

1 Preface

1.1 Brief product description

The Universal Database System UDS/SQL is a high-performance database system based on the structural concept of CODASYL. Its capabilities, however, go far beyond those of CODASYL as it also offers the features of the relational model. Both models can be used in coexistence with each other on the same data resources.

COBOL DML, CALL DML and (ISO standard) SQL are available for querying and updating data. COBOL DML statements are integrated in the COBOL language, CALL DML can be called from any programming language, and SQL statements can be embedded in programs written in COBOL or be used within DRIVE programs.

To ensure confidentiality, integrity and availability, UDS/SQL provides effective but flexible protection mechanisms that control access to the database. These mechanisms are compatible with the openUTM transaction monitor.

The data security concept provided by UDS/SQL effectively protects data against corruption and loss. This concept combines UDS/SQL-specific mechanisms such as logging updated information with BS2000 functions such as DRV (Dual Recording by Volume).

If the add-on product UDS-D is used, it is also possible to process data resources in BS2000 computer networks. UDS/SQL ensures that the data remains consistent throughout the network. Distributed transaction processing in both BS2000 computer networks and networks of BS2000 and other operating systems can be implemented using UDS/SQL together with openUTM-D or openUTM (Unix/Linux/Windows). UDS/SQL can also be used as the database in client-server solutions via SQL gateway or ODBC servers.

The architecture of UDS/SQL (e.g. multitasking, multithreading, DB cache) and its structuring flexibility provide a very high level of throughput.
1.2 Target group

This manual is intended for the database administrator, i.e. the person responsible for updating and reconstructing databases, reorganizing data, and checking databases for consistency.

The database administrator must be familiar with all the steps involved in creating a database (database design, schema, subschema, and SSL generation) and must know how to write DB application programs.

In addition, the DB administrator should have a comprehensive knowledge of BS2000, be familiar with the UDS/SQL transaction concept and the general security concept of UDS/SQL (see the manual “Database Operation”), and also be thoroughly acquainted with the files of a UDS/SQL database and the UDS/SQL utility routines (see the manual “Creation and Restructuring”, Files and realms of a UDS/SQL database).
1.3 Summary of contents

What does this manual contain?

This manual describes all the administrative and operational activities necessary to ensure trouble-free operation of the database. This includes:

– updating and reconstructing the database,
– checking the consistency of the database,
– the output of database information,
– reorganization of the database, and
– controlling the reuse of deallocated database keys.

Illustrative examples are provided to explain these functions.

Using the manuals

The “Guide through the manuals” section below explains which manuals and which parts of the manuals contain the information you require. A glossary gives brief definitions of the technical terms used in the text.

In addition to using the table of contents, you can find answers to your queries either via the index or by referring to the running headers.

Guide through the manuals

The UDS/SQL database system is documented in five manuals:

– UDS/SQL Design and Definition
– UDS/SQL Application Programming
– UDS/SQL Creation and Restructuring
– UDS/SQL Database Operation
– UDS/SQL Recovery, Information and Reorganization

Further manuals describing additional UDS/SQL products and functions are listed on page 7.
For a basic introduction you should refer to chapters 2 and 3 of the “Design and Definition” manual; these chapters describe

– reasons for using databases
– the CODASYL database model
– the relational database model with regard to SQL
– the difference between the models
– the coexistence of the two database models in a UDS/SQL database
– the characteristic features of UDS/SQL

How the manuals are used depends on your previous knowledge and tasks. Table 1 serves as a guide to help you find your way through the manuals.

Examples

If your task is to write COBOL DML programs, you should look up the column “COBOL/CALL DML Programming” under “User task” in the second line of table 1. There, the following chapters of the “Design and Definition” manual are recommended:

General information \( B = \) Basic information
Schema DDL \( D = \) Detailed information
SSL \( D = \) Detailed information
Subschema DDL \( L = \) Learning the functions

In the same column you can also see which chapters of the other manual are of use. Database administrators who are in charge of database administration and operation will find the appropriate information under the column “Administration and Operation”.

Examples
## Preface

<table>
<thead>
<tr>
<th>Contents of the five main manuals</th>
<th>User task</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Design and definition</td>
</tr>
<tr>
<td>Manual UDS/SQL Design and Definition</td>
<td>B</td>
</tr>
<tr>
<td>General information</td>
<td>B</td>
</tr>
<tr>
<td>Designing the database</td>
<td>B</td>
</tr>
<tr>
<td>Schema DDL</td>
<td>L</td>
</tr>
<tr>
<td>SSL</td>
<td>L</td>
</tr>
<tr>
<td>Subschema DDL</td>
<td>L</td>
</tr>
<tr>
<td>Relational schema</td>
<td>L</td>
</tr>
<tr>
<td>Structure of pages</td>
<td>D</td>
</tr>
<tr>
<td>Structure of records and tables</td>
<td>D</td>
</tr>
<tr>
<td>Reference section</td>
<td>S</td>
</tr>
</tbody>
</table>

## Manual UDS/SQL Application Programming

<table>
<thead>
<tr>
<th>Preface</th>
<th>Overview</th>
<th>Transaction concept</th>
<th>Currency table</th>
<th>DML functions</th>
<th>Using DML</th>
<th>COBOL DML reference section</th>
<th>CALL DML reference section</th>
<th>Testing DML functions using DMLTEST</th>
</tr>
</thead>
<tbody>
<tr>
<td>–</td>
<td>B</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>B</td>
<td>B</td>
<td>B</td>
<td>–</td>
</tr>
<tr>
<td>–</td>
<td>B</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>–</td>
<td>L</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>D</td>
<td>–</td>
<td>D</td>
<td>–</td>
</tr>
<tr>
<td>D</td>
<td>L</td>
<td>–</td>
<td>–</td>
<td>L</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>–</td>
<td>L</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>D</td>
<td>–</td>
<td>D</td>
<td>–</td>
</tr>
<tr>
<td>–</td>
<td>L</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>

Table 1: Guide through the manuals (part 1 of 3)
## Manual UDS/SQL Creation and Restructuring

<table>
<thead>
<tr>
<th>Contents of the five main manuals</th>
<th>User task</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Design and definition</td>
</tr>
<tr>
<td>Preface</td>
<td>–</td>
</tr>
<tr>
<td>Overview</td>
<td>–</td>
</tr>
<tr>
<td>Database creation</td>
<td>–</td>
</tr>
<tr>
<td>Defining access rights</td>
<td>–</td>
</tr>
<tr>
<td>Storing and unloading data</td>
<td>D</td>
</tr>
<tr>
<td>Restructuring the database</td>
<td>D</td>
</tr>
<tr>
<td>Database conversion</td>
<td>D</td>
</tr>
<tr>
<td>Version migration with BTRANS24</td>
<td>–</td>
</tr>
</tbody>
</table>

## Manual UDS/SQL Database Operation

| Preface                          | –          | –                         | –               | B                        | B                        | B                | –                |
| The database handler             | –          | –                         | –               | L                        | –                        | –                | D                |
| DBH load parameters              | –          | –                         | –               | L                        | –                        | –                | D                |
| Administration                   | –          | –                         | –               | L                        | –                        | –                | D                |
| High availability                | –          | –                         | –               | L                        | –                        | –                | D                |
| Extending realms during database operation | D          | –                         | –               | B                        | –                        | –                | –                |
| Saving and recovering a database in the event of errors | D          | –                         | –               | D                        | L                        | D                | D                |
| Optimizing performance           | –          | –                         | –               | D                        | –                        | –                | D                |
| Using BS2000 functionality       | –          | –                         | –               | D                        | –                        | –                | –                |
| The SQL conversation             | –          | –                         | –               | L                        | –                        | –                | –                |
| UDSMON                           | –          | –                         | –               | D                        | –                        | –                | –                |
| General functions of the utility routines | –          | –                         | –               | D                        | –                        | –                | –                |
| Using IQS                        | –          | –                         | –               | L                        | D                        | –                | D                |
| Using UDS-D                      | D          | D                         | –               | D                        | D                        | D                | –                |
| Function codes of DML statements | –          | D                         | –               | D                        | –                        | –                | –                |

Table 1: Guide through the manuals

(part 2 of 3)
### Preface

#### Summary of contents

<table>
<thead>
<tr>
<th>Contents of the five main manuals</th>
<th>User task</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Design and definition</td>
</tr>
<tr>
<td><strong>Manual</strong></td>
<td></td>
</tr>
<tr>
<td>UDS/SQL Recovery, Information and Reorganization</td>
<td></td>
</tr>
<tr>
<td>Preface</td>
<td>–</td>
</tr>
<tr>
<td>Updating and reconstructing a database</td>
<td>D</td>
</tr>
<tr>
<td>Checking the consistency of a database</td>
<td>–</td>
</tr>
<tr>
<td>Output of database information</td>
<td>D</td>
</tr>
<tr>
<td>Database reorganization</td>
<td>D</td>
</tr>
<tr>
<td>Controlling the reuse of deallocated database keys</td>
<td>D</td>
</tr>
</tbody>
</table>

#### Additional Manuals

<table>
<thead>
<tr>
<th>Manual</th>
<th>Design and definition</th>
<th>COBOL/CALL DML programming</th>
<th>SQL programming</th>
<th>Creation and restructuring</th>
<th>Administration and operation</th>
<th>Working with openUTM</th>
<th>Working with IQS</th>
<th>Working with UDS-D</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDS/SQL Messages</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>UDS/SQL System Reference Guide</td>
<td>S</td>
<td>S</td>
<td>–</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>IQS</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>D</td>
<td>D</td>
<td>–</td>
<td>L</td>
<td>–</td>
</tr>
<tr>
<td>ADILOS</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>D</td>
<td>–</td>
<td>L</td>
<td>–</td>
</tr>
<tr>
<td>KDBS</td>
<td>–</td>
<td>L¹</td>
<td>–</td>
<td>D</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>SQL for UDS/SQL Language Reference Manual</td>
<td>–</td>
<td>–</td>
<td>D</td>
<td>–</td>
<td>D</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>

| **Table 1: Guide through the manuals** |

<table>
<thead>
<tr>
<th>Manual</th>
<th>Design and definition</th>
<th>COBOL/CALL DML programming</th>
<th>SQL programming</th>
<th>Creation and restructuring</th>
<th>Administration and operation</th>
<th>Working with openUTM</th>
<th>Working with IQS</th>
<th>Working with UDS-D</th>
</tr>
</thead>
<tbody>
<tr>
<td>UDS/SQL Messages</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>UDS/SQL System Reference Guide</td>
<td>S</td>
<td>S</td>
<td>–</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>S</td>
<td>S</td>
</tr>
<tr>
<td>IQS</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>D</td>
<td>D</td>
<td>–</td>
<td>L</td>
<td>–</td>
</tr>
<tr>
<td>ADILOS</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>D</td>
<td>–</td>
<td>L</td>
<td>–</td>
</tr>
<tr>
<td>KDBS</td>
<td>–</td>
<td>L¹</td>
<td>–</td>
<td>D</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td>SQL for UDS/SQL Language Reference Manual</td>
<td>–</td>
<td>–</td>
<td>D</td>
<td>–</td>
<td>D</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>

1 only for COBOL-DML

B →→ provides basic information if you have no experience of UDS/SQL
L →→ helps you learn functions
D →→ provides detailed information
S →→ provides a reference to syntax rules for practical work with UDS/SQL
Additional notes on the manuals

References to other manuals appear in abbreviated form. For example:
(see the “Application Programming” manual, CONNECT)
advise you to look up CONNECT in the “Application Programming” manual.
The complete titles of the manuals can be found under “Related publications” at the back of the manual.

UDS/SQL Messages

This manual contains all messages output by UDS/SQL. The messages are sorted in ascending numerical order, or in alphabetical order for some utility routines.

UDS/SQL System Reference Guide

The UDS/SQL System Reference Guide gives an overview of the UDS/SQL functions and formats.

SQL for UDS/SQL
Language Reference Manual

This manual describes the SQL DML language elements of UDS/SQL.
In addition to UDS/SQL-specific extensions, the language elements described include dynamic SQL as an essential extension of the SQL standard.

1.4 README file

Information on functional changes and additions to the current product version described in this manual can be found in the product-specific README file. You will find the README file on your BS2000 computer under the file name SYSRME.product.version.language. The user ID under which the README file is cataloged can be obtained from your systems support staff. You can view the README file using the /SHOW-FILE command or an editor, and print it out on a standard printer using the following command:

/PRINT-DOCUMENT filename, LINE-SPACING=*BY-EBCDIC-CONTROL
1.5 Changes since the last version

The main changes introduced in UDS/SQL V2.4 in comparison with Version 2.3 are listed in table 2 below, together with the manuals and the chapters in which the changes are described. If a specific topic has been dealt with in more than one manual, the manual in which a detailed description appears is listed first. The following codes are used in the “Manual” column for the individual manuals involved:

DES Design and Definition  DBO Database Operation
APP Application Programming  RIR Recovery, Information and Reorganization
CRE Creation and Restructuring  MSG Messages

<table>
<thead>
<tr>
<th>Topic</th>
<th>Manual</th>
<th>Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Online DBTT extension</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>The DBTT for a record type in the user schema can also be extended either automatically or manually within a DBH user session without interrupting transaction operation</td>
<td>DBO</td>
<td>6</td>
</tr>
<tr>
<td><strong>New DAL commands</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACT DBTT-INCR - Activate online extensibility for a DBTT</td>
<td>DBO</td>
<td>4</td>
</tr>
<tr>
<td>DEACT DBTT-INCR - Deactivate online extensibility for a DBTT</td>
<td>DBO</td>
<td>4</td>
</tr>
<tr>
<td>DISPLAY DBTT-INCR - List information on a DBTT</td>
<td>DBO</td>
<td>4</td>
</tr>
<tr>
<td>EXTEND DBTT - Perform online DBTT extension</td>
<td>DBO</td>
<td>4</td>
</tr>
<tr>
<td><strong>Modified DAL commands</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ACT INCR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Notes on online DBTT extension included</td>
<td>DBO</td>
<td>4</td>
</tr>
<tr>
<td>DEACT INCR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Notes on online DBTT extension included</td>
<td>DBO</td>
<td>4</td>
</tr>
<tr>
<td>DISPLAY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>New LINES parameter in DISPLAY SUBSCH, DISPLAY REALMS, DISPLAY FPA and DISPLAY INCR</td>
<td>DBO</td>
<td>4</td>
</tr>
<tr>
<td>DISPLAY INCR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>New FMT column informs you whether formatting of the newly added database pages by the DBH is necessary</td>
<td>DBO</td>
<td>4</td>
</tr>
<tr>
<td>REACT INCR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Notes on online DBTT extension included</td>
<td>DBO</td>
<td>4</td>
</tr>
</tbody>
</table>

Table 2: Changes since the last version (part 1 of 3)
<table>
<thead>
<tr>
<th>Topic</th>
<th>Manual</th>
<th>Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>BALTER utility routine</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Modified runtime messages due to online DBTT extension</td>
<td>CRE</td>
<td>6</td>
</tr>
<tr>
<td><strong>BCHECK utility routine</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>New and modified runtime and inconsistency messages due to online DBTT extension</td>
<td>RIR</td>
<td>3</td>
</tr>
<tr>
<td>It is now possible to evaluate job switches set by BCHECK.</td>
<td>RIR</td>
<td>3</td>
</tr>
<tr>
<td><strong>BPGSIZE utility routine</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>In the new database each DBTT consists only of a DBTT base, DBTT extents are eliminated.</td>
<td>CRE</td>
<td>7</td>
</tr>
<tr>
<td><strong>BPRECORD utility routine</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>New “IN CSV” parameter for outputting data in CSV format</td>
<td>RIR</td>
<td>7</td>
</tr>
<tr>
<td><strong>BPSIA utility routine</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>The table DBTT INFORMATION has been modified: “BEGIN” column has been renamed to “ANCHOR” and contains the first page of the DBTT anchor table New “EXTENTS” column inserted</td>
<td>RIR</td>
<td>4</td>
</tr>
<tr>
<td>New “IN CSV” parameter for outputting data in CSV format</td>
<td>RIR</td>
<td>4</td>
</tr>
<tr>
<td><strong>BREORG utility routine</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>The MODIFY-RECORD-POPULATION statement outputs detailed information on DBTT changes</td>
<td>RIR</td>
<td>8</td>
</tr>
<tr>
<td><strong>BSTATUS utility routine</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>New columns inserted in the record type statistics: EXTENDIBLE, NR OF DBTT EXTENTS, HIGHEST USED EXTENT</td>
<td>RIR</td>
<td>6</td>
</tr>
<tr>
<td>New “IN CSV” parameter for outputting data in CSV format</td>
<td>RIR</td>
<td>6</td>
</tr>
<tr>
<td><strong>BTRANS24 utility routine</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>New utility routine for version migration to UDS/SQL V2.4</td>
<td>CRE</td>
<td>8</td>
</tr>
<tr>
<td><strong>Output of database information in system independent format</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data can be output in CSV format for simplified further processing in other system environments</td>
<td>DBO</td>
<td>12</td>
</tr>
<tr>
<td>New “IN CSV” parameter in the BPSIA, BSTATUS and BPRECORD utility routines</td>
<td>RIR</td>
<td>4, 6, 7</td>
</tr>
<tr>
<td><strong>&quot;STATUS&quot; screen of UDS/SQL monitor</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>The TSN column also shows for UTM applications the TSN of the last UTM task used in the transaction</td>
<td>DBO</td>
<td>11</td>
</tr>
</tbody>
</table>

Table 2: Changes since the last version (part 2 of 3)
<table>
<thead>
<tr>
<th>Topic</th>
<th>Manual</th>
<th>Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>New information available concerning realm formatting</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>You can use the DAL command DISPLAY INCR to find out whether the newly added database pages will be formatted in the case of an online realm extension</td>
<td>DBO</td>
<td>6, 4</td>
</tr>
<tr>
<td><strong>Support for files larger than 32 GB</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UDS/SQL supports files having the LARGE FILE property</td>
<td>DBO CRE</td>
<td>6 2</td>
</tr>
<tr>
<td><strong>MPVS used for UDS/SQL</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Note extended to include public volume sets in the QUIET state.</td>
<td>DBO</td>
<td>9.3</td>
</tr>
<tr>
<td><strong>Online backups</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>It is possible to use HSMS as well as ARCHIVE to perform online backups.</td>
<td>DBO</td>
<td>7, 9</td>
</tr>
<tr>
<td>If online realm extension is active, online backups must be created with HSMS or ARCHIVE.</td>
<td>RIR DBO</td>
<td>2 7</td>
</tr>
<tr>
<td><strong>Data input via SYSDTA for DDL and SSL compilers</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>If input is performed via SYSDTA, then the input stream must at least contain the END statement</td>
<td>CRE</td>
<td>3</td>
</tr>
<tr>
<td><strong>Administration of UDS/SQL via DCAM</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extended DCAM administration of the DBH in an OMNIS environment</td>
<td>DBO</td>
<td>4</td>
</tr>
<tr>
<td><strong>Page structure</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description of the Act-Key-0 and Act-Key-N page modified</td>
<td>DES</td>
<td>7</td>
</tr>
<tr>
<td>Description of the DBTT anchor page inserted</td>
<td>DES</td>
<td>7</td>
</tr>
<tr>
<td><strong>New or modified messages</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UDS0410, UDS0738, UDS0741, UDS0742, UDS0743, UDS0744, UDS0745, UDS0209, UDS0722, UDS0841</td>
<td>MSG</td>
<td>2</td>
</tr>
<tr>
<td>0900, 0389, 0505, 1061, 1062, 1403, 1419, 1481, 2403, 2404, 2406, 2507, 2551, 2552, 3041, 3609, 3666, 3667, 3668 omitted, 5011, 6415, 6416,</td>
<td>MSG</td>
<td>3</td>
</tr>
<tr>
<td>New messages for the BTRANS24 utility routine inserted</td>
<td>MSG</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 2: Changes since the last version (part 3 of 3)
1.6 Notational conventions

This section provides an explanation of the symbols used for warnings and notes as well as the notational conventions used to describe syntax rules.

1.6.1 Warnings and notes

| **i** | Points out particularly important information |
| **⚠️** | CAUTION! | Warnings |

1.6.2 NON-SDF notational conventions

<table>
<thead>
<tr>
<th>Language element</th>
<th>Explanation</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>KEYWORD</strong></td>
<td>Keywords are shown in underlined uppercase letters. You may enter either the underlined parts of the keyword exactly as shown or the complete keyword.</td>
<td>DATABASE-KEY, MANUAL</td>
</tr>
<tr>
<td><strong>OPTIONAL WORD</strong></td>
<td>Optional words are shown in uppercase letters without underlining. Such words may be omitted without altering the meaning of a statement.</td>
<td>NAME IS, ALLOWED, PAGES</td>
</tr>
<tr>
<td>variable</td>
<td>Variables are shown in italic lowercase letters. In a format which contains variables, a current value must be entered in place of each variable.</td>
<td>item-name, literal-3, integer</td>
</tr>
<tr>
<td>{Either}</td>
<td>Exactly one of the expressions enclosed in braces must be specified. Indented lines belong to the preceding expression. The braces themselves must not be specified.</td>
<td>{CALC, INDEX}, {VALUE IS, VALUES ARE}</td>
</tr>
<tr>
<td>[optional]</td>
<td>The expression in square brackets can be omitted. UDS/SQL then uses the default value. The brackets themselves must no be specified.</td>
<td>[IS integer], [WITHIN realm-name]</td>
</tr>
</tbody>
</table>

Table 3: Notational conventions (part 1 of 2)
All other characters such as ( ) , . ; “ = are not metacharacters: they must be specified exactly as they appear in the formats.

<table>
<thead>
<tr>
<th>Language element</th>
<th>Explanation</th>
<th>Example</th>
</tr>
</thead>
</table>
| ... or ....      | The immediately preceding expression can be repeated several times if required. The two language elements distinguish between repetitions which use blanks and those which use commas. | item-name,...
|                  |             | {SEARCH KEY.....}... |
| . . . .          | Indicates where entries have been omitted for reasons of clarity. When the formats are used, these omissions are not allowed. | SEARCH KEY IS ..... RECORD NAME . . . |
|                  | The period must be specified and must be followed by at least one blank. The underline must not be specified. | SET SECTION. 03 item-name..... . |
| Space            | Means that at least one blank has to be specified. | USING CALC |

Table 3: Notational conventions (part 2 of 2)
1.6.3 SDF syntax representation

This syntax description is based on SDF Version 4. The syntax of the SDF command/statement language is explained in the following three tables.

**table 4: Metasyntax**

Certain characters and representations are used in the statement formats; their meaning is explained in table 4.

**table 5: Data types**

Variable operand values are represented in SDF by data types. Each data type represents a specific value set. The number of data types is limited to those described in table 5. The description of the data types is valid for all commands and statements. Therefore only deviations from table 5 are explained in the relevant operand descriptions.

**table 6: Data type suffixes**

The description of the “integer” data type in table 6 also contains a number of items in italics. The italics are not part of the syntax, but are used merely to make the table easier to read.

The description of the data type suffixes is valid for all commands and statements. Therefore only deviations from table 6 are explained in the relevant operand descriptions.

<table>
<thead>
<tr>
<th>Representation</th>
<th>Meaning</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>UPPERCASE LETTERS</td>
<td>Uppercase letters denote keywords. Some keywords begin with *.</td>
<td>OPEN DATABASE</td>
</tr>
<tr>
<td></td>
<td></td>
<td>COPY-NAME = *NONE</td>
</tr>
<tr>
<td>=</td>
<td>The equal sign connects an operand name with the associated operand values.</td>
<td>CONFIGURATION-NAME = &lt;name 1..8&gt;</td>
</tr>
<tr>
<td>&lt; &gt;</td>
<td>Angle brackets denote variables whose range of values is described by data types and their suffixes (Tables 5 and 6).</td>
<td>DATABASE = &lt;dbname&gt;</td>
</tr>
<tr>
<td>Underscoring</td>
<td>Underscoring denotes the default value of an operand.</td>
<td>SCHEMA-NAME = *STD</td>
</tr>
<tr>
<td>/</td>
<td>A slash separates alternative operand values.</td>
<td>CMD = *ALL / &lt;dal-cmd&gt;</td>
</tr>
</tbody>
</table>

Table 4: Metasyntax
### SDF syntax representation

<table>
<thead>
<tr>
<th>Representation</th>
<th>Meaning</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>(...)</td>
<td>Parentheses denote operand values that initiate a structure.</td>
<td>*KSET-FORMAT(…)</td>
</tr>
</tbody>
</table>

**Indentation**

Indentation indicates that the operand is dependent on a higher-ranking operand.

| USER-GROUP-NAME = *KSET-FORMAT(…)
| *KSET-FORMAT(…)
| HOST = <host> |

A vertical bar identifies related operands within structure. Its length marks the beginning and end of a structure. A structure may contain further structures. The number of vertical preceding an operand corresponds to the depth of the structure.

| USER-GROUP-NAME = *ALL-EXCEPT(…)
| *ALL-EXCEPT(…)
| NAME = *KSET-FORMAT(…)
| *KSET-FORMAT(…)
| HOST = <host>
| … |

A comma precedes further operands at the same structure level.

| ,NAME = STD |

*list-poss(n):* signifies that the operand values following it may be entered as a list. If a value is specified for (n), the list may contain no more than that number of elements. A list of two or more elements must be enclosed in parentheses.

| NAME = list-poss(30): <subschema-name> |

---

Table 4: Metasyntax (part 2 of 2)
<table>
<thead>
<tr>
<th>Data type</th>
<th>Character set</th>
<th>Special rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>alog-seq-no</td>
<td>0..9</td>
<td>1..9 characters</td>
</tr>
</tbody>
</table>
| appl        | A..Z 0..9 $,#,@ | 1..8 characters  
String that can consist of a number of substrings separated by hyphens; first character A..Z or $, #, @  
Strings of less than 8 characters are filled internally with underscore characters. |
|catid        | A..Z 0..9     | 1..4 characters  
must not start with the string PUB |
|copy-name    | A..Z 0..9     | 1..7 characters, starting with A..Z |
|c-string     | EBCDIC characters | 1..4 characters  
Must be enclosed in single quotes; the letter C may be used as a prefix.  
Single quotes within c-string must be specified twice. |
|csv-filename | A..Z 0..9     | 1..30 characters  
Must be enclosed in single quotes |
|dal-cmd      | A..Z 0..9     | 1..64 characters |
|date         | 0..9          | Date specification  
Input format: yyyy-mm-dd  
yyyy : year; may be 2 or 4 digits long  
mm : month  
dd : day |
|dbname       | A..Z 0..9     | 1..17 characters, starting with A..Z |
|device       | A..Z 0..9 $,#,@ | 5..8 characters, starting with A..Z or 0..9  
String that can consist of a number of substrings separated by hyphens and which corresponds to a device. In the dialog guidance, SDF shows the permissible operand values. Information as to the possible devices can be found in the relevant operand description. |

Table 5: Data types (part 1 of 3)
### Data type | Character set | Special rules |
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>host</td>
<td>A..Z 0..9 $,#,@</td>
<td>1..8 characters</td>
</tr>
<tr>
<td>integer</td>
<td>0..9,+,-</td>
<td>+ or - may only be the first character.</td>
</tr>
<tr>
<td>kset</td>
<td>A..Z 0..9 $,#,@</td>
<td>1..8 characters</td>
</tr>
<tr>
<td>name</td>
<td>A..Z 0..9 $,#,@</td>
<td>1..8 characters</td>
</tr>
<tr>
<td>realm-name</td>
<td>A..Z 0..9 $,#,@</td>
<td>1..30 characters</td>
</tr>
<tr>
<td>realmref</td>
<td>0..9</td>
<td>1..3 characters</td>
</tr>
<tr>
<td>record-name</td>
<td>A..Z 0..9 $,#,@</td>
<td>1..30 characters</td>
</tr>
<tr>
<td>recordref</td>
<td>0..9</td>
<td>1..3 characters</td>
</tr>
<tr>
<td>schema-name</td>
<td>A..Z 0..9 $,#,@</td>
<td>1..30 characters</td>
</tr>
<tr>
<td>set-name</td>
<td>A..Z 0..9 $,#,@</td>
<td>1..30 characters</td>
</tr>
</tbody>
</table>

Table 5: Data types (part 2 of 3)
### Data type

<table>
<thead>
<tr>
<th>Data type</th>
<th>Character set</th>
<th>Special rules</th>
</tr>
</thead>
<tbody>
<tr>
<td>subschema-name</td>
<td>A..Z 0..9</td>
<td>1..30 characters String that can consist of a number of substrings separated by hyphens; first character: A..Z</td>
</tr>
<tr>
<td></td>
<td>Structure identifier: hyphen</td>
<td></td>
</tr>
<tr>
<td>time</td>
<td>0..9</td>
<td>Time-of-day specification</td>
</tr>
<tr>
<td></td>
<td>Structure identifier: colon</td>
<td></td>
</tr>
<tr>
<td>userid</td>
<td>A..Z 0..9 $,#,@</td>
<td>1..8 characters, beginning with A..Z or $,#,@ BPRIVACY: Strings of less than 8 characters are filled internally with underscore characters.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>volume</td>
<td>A..Z 0..9 $,#,@</td>
<td>1..6 characters starting with A..Z or 0..9</td>
</tr>
<tr>
<td>x-string</td>
<td>Hexadecimal: 00..FF</td>
<td>1..8 characters Must be enclosed in single quotes and prefixed with the letter X. There may be an odd number of characters</td>
</tr>
</tbody>
</table>

Table 5: Data types (part 3 of 3)

### Suffix

<table>
<thead>
<tr>
<th>Suffix</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>x..y unit</td>
<td>For the “integer” data type: range specification. Minimum value permitted for “integer”. x is an (optionally signed) integer. Maximum value permitted for “integer”. y is an (optionally signed) integer. for “integer” only: additional units. The following units may be specified: Mbyte, Kbyte, seconds.</td>
</tr>
</tbody>
</table>

Table 6: Data type suffixes
1.7 Sample databases

The SHIPPING and CUSTOMER databases form the basis for most of the examples and utility routines in this manual.

Figure 1: SHIPPING database with schema name MAIL-ORDERS
Sample databases

Figure 2: CUSTOMER database with schema name CUSTOMER-FILE

Figure 3: PERSONAL database with schema name PERS-DB
2 Updating and reconstructing a database with BMEND

The BMEND utility routine performs the function of updating and reconstructing a database within the general framework of the UDS/SQL security concept (see the “Database Operation” manual).

BMEND can be used to update the entire database or to update individual or selected realms with one or more ALOG files. It can be applied on both the original database and the shadow database as well as detached realms.

The database can be updated using ALOG files with the UPDATE-DATABASE statement.

The BMEND statement SHOW-LOG-INFORMATION shows which ALOG files must be specified when updating individual realms in order to make their state consistent with that of the entire database. In other words, you must supply these ALOG files for the update. You are also provided with information on whether a logging gap exists in the sequence of ALOG files and whether any ALOG file was terminated in an inconsistent state. In such cases, updates can only be applied until that point in time. Later it is only possible using backup.
2.1 Functions

The BMEND utility routine provides the following functions for

– updating databases and shadow databases (or specific realms thereof)
– attaching (ADD) and detaching (REMOVE) realms
– enabling and disabling the online-copy facility for a database
– starting and stopping AFIM logging
– obtaining information on the status of databases, shadow databases and ALOG files, and
– concurrently updating detached realms in parallel with DBH operations and calling an information function for realms or ALOG files
2.2 Statements

2.2.1 Rules

Incorrectly entered statements can be corrected.

Multiple statements of the same type (except for SHOW-LOG-INFORMATION) are combined and only executed once.

If conflicting specifications concerning the function or object are made (e.g. START-LOG/STOP-LOG or ADD-REALM/REMOVE-REALM), the last specification entered applies.

All valid statements, except for the ALLOCATE-BUFFER-POOL, OPEN-DATABASE and UNDO statements, are executed after the END statement:

- The ALLOCATE-BUFFER-POOL statement must be the first statement specified.
- The OPEN-DATABASE statement is only permitted if no ADD-FILE-LINK LINK-NAME=DATABASE has been specified.
- Every correctly entered statement can be reversed with the UNDO statement or with its inverse function (if one exists)

Statements may be entered in any order. Execution occurs in the following order:

- ALLOCATE-BUFFER-POOL
- OPEN-DATABASE
- UPDATE-DATABASE
- ADD-REALM
- REMOVE-REALM
- START-LOG
- STOP-LOG
- KILL-LOG
- ENABLE-ONLINE-COPY
- DISABLE-ONLINE-COPY
- SHOW-LOG-INFORMATION
2.2.2 Permitted functions

The range of functions permitted depends on the following questions:

– Is the original database or shadow database involved?
– Is the database or shadow database being processed by the DBH?
– Is the database consistent?
– Has AFIM logging been enabled?

Functions that are not permitted are not shown in the SDF mask.

The set of functions permitted may change in the course of a BMEND run, since BMEND statements can have an effect on consistency and logging.

Consistency point information can always be output at any time.

The KILL-LOG statement is only permitted for inconsistent databases. This statement is required for a database warm start without logging.

List of special cases

– Original database with active DBH:
  UPDATE-DATABASE
  Concurrent updating with a session is only allowed for realms that are detached according to DBDIR-AK0.

– Inconsistent original database with logging and without active DBH:
  ADD-REALM
  REMOVE-REALM
  START-LOG
  STOP-LOG
  Rejected if not preceded by an UPDATE-DATABASE statement to create a consistent database.

– Consistent original database without logging and without active DBH:
  ENABLE-ONLINE-COPY
  Only permitted if the START-LOG statement has already been issued

– Shadow database with active DBH:
  UPDATE-DATABASE
  Concurrent updating with a session is only allowed for realms that are detached according to DBDIR-AK0.
Inconsistent shadow database without active DBH:

ADD-REALM
REMOVE-REALM

Rejected if not preceded by an UPDATE-DATABASE statement to create a consistent database.

2.2.3 BMEND statements

<table>
<thead>
<tr>
<th>Statement</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADD-REALM</td>
<td>Attaches realms to a database</td>
</tr>
<tr>
<td>ALLOCATE-BUFFER-POOL</td>
<td>Defines the buffer size</td>
</tr>
<tr>
<td>DISABLE-ONLINE-COPY</td>
<td>Disables the online-copy facility for the database</td>
</tr>
<tr>
<td>ENABLE-ONLINE-COPY</td>
<td>Enables the online-copy facility for the database</td>
</tr>
<tr>
<td>END</td>
<td>Terminates the input of statements</td>
</tr>
<tr>
<td>KILL-LOG</td>
<td>Stops logging for an inconsistent database</td>
</tr>
<tr>
<td>OPEN-DATABASE</td>
<td>Opens the database</td>
</tr>
<tr>
<td>REMOVE-REALM</td>
<td>Detaches realms</td>
</tr>
<tr>
<td>SHOW-LOG-INFORMATION</td>
<td>Displays logging information</td>
</tr>
<tr>
<td>START-LOG</td>
<td>Starts logging for the database original</td>
</tr>
<tr>
<td>STOP-LOG</td>
<td>Stops logging for database operations</td>
</tr>
<tr>
<td>UNDO</td>
<td>Cancels the effect of a statement</td>
</tr>
<tr>
<td>UPDATE-DATABASE</td>
<td>Applies AFIMs to the database</td>
</tr>
</tbody>
</table>

Table 7: BMEND statements

The individual statements of BMEND are described below in alphabetical order.
Attach realms to a database (ADD-REALM)

The ADD-REALM statement can be used to attach one or more realms to a database. The realms to be attached must be consistent and compatible with the current DBDIR.

<table>
<thead>
<tr>
<th>ADD-REALM</th>
</tr>
</thead>
<tbody>
<tr>
<td>REALM-NAME = *ALL / *ALL-EXCEPT(...) / list-poss(30): &lt;realm-name&gt;</td>
</tr>
<tr>
<td>*ALL-EXCEPT(...)</td>
</tr>
<tr>
<td>NAME = list-poss(30): &lt;realm-name&gt;</td>
</tr>
</tbody>
</table>

**REALM-NAME = *ALL**

All detached realms are attached.

**REALM-NAME = *ALL-EXCEPT(...)**

All detached realms except for those specified are attached.

    NAME = list-poss(30): <realm-name>

Name(s) of the realm(s) to be excluded.

**REALM-NAME = list-poss(30): <realm-name>**

All specified realms are attached.

The realms DBDIR and DBCOM cannot be attached and are therefore rejected.
Define buffer size (ALLOCATE-BUFFER-POOL)

The ALLOCATE-BUFFER-POOL statement defines the size of the used buffer pool in Mbytes.

This statement may be optionally omitted (if the default value is desired); otherwise, it must be specified as the first statement.

ALLOCATE-BUFFER-POOL

| BUFFER-SIZE = STD / <integer 1..2000> |

BUFFER-SIZE = STD
The standard size of the buffer pool is defined as 1 Mbyte.

BUFFER-SIZE = <integer 1..2000>
The size of the buffer pool must lie within the given limits. The maximum value depends on the version of the operating system and the configuration of working memory in the system.

The value for ADDRESS-SPACE-LIMIT must be greater than the value specified here. The appropriate value can be set by the system administrator with the MODIFY-USER-ATTRIBUTES command.
Disable online-copy facility for the database (DISABLE-ONLINE-COPY)

The DISABLE-ONLINE-COPY statement disables the online-copy facility for all realms of the database.

This statement is also permitted for a shadow database.

<table>
<thead>
<tr>
<th>DISABLE-ONLINE-COPY</th>
</tr>
</thead>
</table>

This statement has no operands.
Enable online-copy facility for the database (ENABLE-ONLINE-COPY)

The ENABLE-ONLINE-COPY statement enables the online-copy facility for all realms of the database.
Prerequisite: AFIM logging must be active.

This statement is also permitted for a shadow database.

The online-copy facility for a database cannot be enabled for an original unless the logging function is active or the START-LOG statement is issued beforehand. Since a consistent database is a prerequisite for the START-LOG statement, an UPDATE-DATABASE statement may also be required.

To create online backups of databases or individual realms, you must use the corresponding HSMS or ARCHIVE statements. In particular if online realm extension is active, you should not create online backups with COPY-FILE since the backup (following a successfully concluded online realm extension) may not contain all the relevant pages of the realm in question and BMEND may therefore no longer be able to generate a consistent state.

The online backup capability of a database is recorded both in the UDS administration data and in the DMS catalog entries for the database files. When a database is copied either with COPY-FILE or with HSMS/ARCHIVE, the specifications relating to the online backup capability in the DMS catalog may be lost depending on the selected parameters. Therefore, before using a copy of a database from an original for which the online backup capability was active, you must ensure that this property remains consistent, for example by running the BMEND utility routine again with the statement ENABLE-ONLINE-COPY.

<table>
<thead>
<tr>
<th>ENABLE-ONLINE-COPY</th>
</tr>
</thead>
</table>

This statement has no operands.
**Terminate input of statements (END)**

The END statement is used to terminate the input of statements. All entered statements are executed after this statement.

The END statement cannot be canceled with the UNDO statement.

```
END
```

This statement has no operands.
Stop logging for an inconsistent database (KILL-LOG)

The KILL-LOG statement is used to suppress AFIM logging in order to perform a database warm start without an ALOG file (due to a hardware error on the ALOG file, for instance). The ALOG file is used in a warm start to include the AFIMs from the RLOG file which are not yet in the ALOG file. The RLOG file is sufficient for a database warm start. The KILL-LOG statement automatically disables the online-copy facility as well.

This statement is only permitted for an inconsistent database.

```
KILL-LOG
```

The KILL-LOG statement has no operands.

When a warm start is performed without the ALOG file, a logging gap is created. It is therefore advisable to save the database after a warm start and to begin logging again before any new changes are made.
Open database (OPEN-DATABASE)

The OPEN-DATABASE statement specifies the database to be processed by the statements which follow.

<table>
<thead>
<tr>
<th>OPEN-DATABASE</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATABASE-NAME = &lt;dbname&gt;</td>
</tr>
<tr>
<td>,COPY-NAME = *NONE / &lt;copy-name&gt;</td>
</tr>
<tr>
<td>,USER-IDENTIFICATION = *OWN / &lt;userid&gt;</td>
</tr>
</tbody>
</table>

**DATABASE-NAME = <dbname>**
Name of the database. You can only process a database that is cataloged under your own user ID. A database under a foreign user ID can only be processed from the system administrator ID TSOS.

**COPY-NAME = *NONE**
The database original is processed.

**COPY-NAME = <copy-name>**
The shadow database with the specified copy name is processed.

**USER-IDENTIFICATION = *OWN**
The database is located under the user's own user ID.

**USER-IDENTIFICATION = <userid>**
The specification of a foreign user ID is only permitted under the system administrator ID TSOS.

- The OPEN-DATABASE statement is not permitted if the database is assigned using ADD-FILE-LINK LINK-NAME=DBASE.
Detach realms (REMOVE-REALM)

The REMOVE-REALM statement can be used to detach consistent realms.

<table>
<thead>
<tr>
<th>REMOVE-REALM</th>
</tr>
</thead>
<tbody>
<tr>
<td>REALM-NAME = *ALL-EXCEPT(...) / list-poss(30): &lt;realm-name&gt;</td>
</tr>
<tr>
<td>*ALL-EXCEPT(...)</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

REALM-NAME = *ALL-EXCEPT(...)  
All attached realms except for those specified are detached.

   Name = list-poss(30): <realm-name>  
   Name(s) of the realms that are not to be detached.

REALM-NAME = list-poss(30): <realm-name>  
All specified realms are detached.

The realms DBDIR and DBCOM cannot be detached and are therefore rejected.
Show log information (SHOW-LOG-INFORMATION)

The SHOW-LOG-INFORMATION statement can be used to output the following information:

- status of realms of the assigned database with respect to the log interval required in order to reach a common consistency point
- attributes of the database “with logging or without logging” and whether “online copies are allowed"
- information on up to 63 ALOG files (history, sequence numbers in ascending order)

The displayed times reflect the local time.

In addition, useful information for the application of updates is stored in a job variable, provided such a job variable has been created with LINK-NAME=*JVBMEND (see section “Supplying job variables” on page 51).

The SHOW-LOG-INFORMATION statement can be specified alone and may also be entered in the course of DBH operations.

```
SHOW-LOG-INFORMATION

    REALM-NAME = *ALL / *ALL-EXCEPT(...) / list-poss(30): <realm-name>
        *ALL-EXCEPT(...)                   
        NAME = list-poss(30): <realm-name>
    LOG-FILE = STD / NONE / <alog-seq-no>
    OUTPUT = list-poss: SYSLST / SYSOUT
```

**REALM-NAME = *ALL**
Shows information for all realms of the assigned database.

**REALM-NAME = *ALL-EXCEPT(...)**
Shows information for all realms except for those specified.

```
    NAME = list-poss(30): <realm-name>
        Name(s) of the realm(s) for which no information is to be shown.
```

**REALM-NAME = list-poss(30): <realm-name>**
Shows information for all specified realms.

**LOG-FILE = STD**
Shows information as of the current ALOG file, i.e. the one specified in the DBDIR.
SHOW-LOG-INFORMATION statement

LOG-FILE = NONE
No information on ALOG files is output.

LOG-FILE = <alog-seq-no>
Shows information as of the specified ALOG file; if the time is not unique (due to conversion from summer time to winter time, for example), a warning is issued.

OUTPUT = list-poss: SYSLST / SYSOUT
Defines where the information is to be output.

SYSLST
Log information is output to SYSLST.

SYSOUT
Log information is output to SYSOUT.

Message texts
The following information is output:

1. Details with respect to the processed database

 ***** LOG INFORMATION FOR DATABASE $userid.dbname[.copy-name]

dbname
Name of the assigned database

copy-name
is output if a shadow database was assigned

2. List of specified realms indicating the ALOG SEQ NR (e.g. with differing ALOG sequence numbers at the start and end of an online copy)

 ***** LOG INTERVAL OF SPECIFIED REALMS

<table>
<thead>
<tr>
<th>REALM NAME</th>
<th>ALOG SEQ NR BEGIN</th>
<th>ALOG SEQ NR END</th>
<th>CONSISTENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>realm-name-1</td>
<td>alog-seq-no</td>
<td>alog-seq-no</td>
<td>YES/NO</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>realm-name-n</td>
<td>alog-seq-no</td>
<td>alog-seq-no</td>
<td>YES/NO</td>
</tr>
</tbody>
</table>

realm-name
Names of the specified realms
SHOW-LOG-INFORMATION statement

ALOG SEQ NR
BEGIN: alog-seq-no of the realm at the start of the backup operation (of the online-copy)
END: alog-seq-no of the realm at the end of the backup operation (of the online-copy)

CONSISTENT
YES: The consistency point date at the start and end of the backup operation is the same, and no SYSTEM BREAK is set.
NO: The status of the realm differs from that of a consistency point (the consistency date at the start and end of the backup operation differ, or a SYSTEM BREAK is set)

3. Specification of the log interval required to apply the update

***** TO MAKE THE SPECIFIED REALMS CONSISTENT, THE FOLLOWING LOG FILES ARE NECESSARY:
FROM ALOG SEQ NR alog-seq-no1 TO ALOG SEQ NR alog-seq-no2
OR FROM LOG INTERVAL BEGIN datetime-1 TO LOG INTERVAL END datetime-2

alog-seq-no1
The update must be applied starting with this ALOG SEQ NR. This value is saved in a job variable.

alog-seq-no2
This value represents the DEADLINE the must be given in order to make all specified realms consistent.

datetime-1
Date and time of the first log

datetime-2
Date and time of the last log

If no updates need to be applied, this output is suppressed.

4. Log mode with volume information; Message 4 is not output for shadow databases.

***** SUPPORTS OF ACTUAL LOG FILE:
DEFAULT SUPPORT:{VOLUME = vol-1 DEVICE = dev-1
[...] /
PVS ID = {DEFAULT PVS / catid}}
RESERVE SUPPORT:{VOLUME = vol-4 DEVICE = dev-4
[...] /
PVS ID = {DEFAULT PVS / catid}}
5. Logging history

***** INFORMATION ABOUT LOG HISTORY:

<table>
<thead>
<tr>
<th>ALOG SEQ NR</th>
<th>LOG INTERVAL</th>
<th>AFIM</th>
<th>BACKOUT</th>
<th>LOGGING</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BEGIN</td>
<td>END</td>
<td>BFIM</td>
<td>GAP</td>
</tr>
</tbody>
</table>

| alog-seq-nr | time-1     | time-2 | [*] | [*] | [{*/?}] |

ALOG SEQ NR
Starting with the sequence number specified via the LOG-FILE operand, information on a maximum of 63 ALOG files is output (with sequence numbers in descending order)

LOG INTERVAL
BEGIN: time-1 Time of the first log
END: time-2 Time of the last log
If the time is not unique, daylight saving time (summer time) is assumed, and a warning is issued.

AFIM *: AFIMs are recorded in the ALOG file
.: Gap in the AFIM log

BACKOUT BFIM
*: BFIMs are recorded in the ALOG file
.: No BFIMs are recorded in the ALOG file

LOGGING GAP
*: The LOG INTERVAL END is less than the LOG INTERVAL BEGIN of the next ALOG file in the sequence or less than the BACK UP DATA of the DBDIR (LOG-FILE=STD)
?: The continuation of logging in the ALOG file with the next sequence number cannot be examined (LOG-FILE=alog-seq-no)
.: Logging was continued to the next ALOG file in the sequence without interruption

If the ALOG file that is specified in the LOG-FILE operand cannot be read, an error message is output instead of the table.

6. Whether online copies are allowed

***** ONLINE COPIES BY ARCHIVE ARE [NOT] ALLOWED
SHOW-LOG-INFORMATION statement

Example

/ASSIGN-SYSDTA TO-FILE=*SYSCMD
/ADD-FILE LINK-NAME=DATABASE,FILE-NAME=VERSAND.DBDIR
/SELECT-PRODUCT VERSION PRODUCT-NAME=UDS-SQL,VERSION=02.4A00
/START-UDS-BMEND

****** START BMEND (UDS/SQL V2.4 0400) 2004-08-12 15:41:32

//SHOW-LOG-INFORMATION REALM-NAME=*ALL,LOG-FILE=4,OUTPUT=SYSOUT

SYSTEM_BREAK OCCURRED IN REALM DATABASE-DIRECTORY

****** INCONSISTENT DATABASE DIRECTORY

SYSTEM_BREAK OCCURRED IN REALM PURCHASE-ORDER-RLM

SYSTEM_BREAK OCCURRED IN REALM FOOD

SYSTEM_BREAK OCCURRED IN REALM ARTICLE-RLM

****** INCONSISTENT DATABASE DIRECTORY

FUNCTION ADD NOT AVAILABLE

FUNCTION REMOVE NOT AVAILABLE

FUNCTION START NOT AVAILABLE

FUNCTION STOP NOT AVAILABLE

//END

****** BEGIN FUNCTION SHOW LOG INFORMATION AT 15:41:32

****** LOG INFORMATION FOR DATABASE $XXXXXXXX.SHIPPING ____________________________________1.

****** LOG INTERVAL OF SPECIFIED REALMS ___________________________________________________2.

<table>
<thead>
<tr>
<th>REALM-NAME</th>
<th>BEGIN</th>
<th>END</th>
<th>CONSISTENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATABASE-DIRECTORY</td>
<td>2</td>
<td>2</td>
<td>NO</td>
</tr>
<tr>
<td>DATABASE-COMPILED-REALM</td>
<td>1</td>
<td>1</td>
<td>YES</td>
</tr>
<tr>
<td>CUSTOMER-ORDER-RLM</td>
<td>1</td>
<td>1</td>
<td>YES</td>
</tr>
<tr>
<td>PURCHASE-ORDER-RLM</td>
<td>2</td>
<td>2</td>
<td>NO</td>
</tr>
<tr>
<td>CLOTHING</td>
<td>1</td>
<td>1</td>
<td>YES</td>
</tr>
<tr>
<td>HOUSEHOLD-GOODS</td>
<td>1</td>
<td>1</td>
<td>YES</td>
</tr>
<tr>
<td>SPORTS-ARTICLES</td>
<td>1</td>
<td>1</td>
<td>YES</td>
</tr>
<tr>
<td>FOOD</td>
<td>2</td>
<td>2</td>
<td>NO</td>
</tr>
<tr>
<td>LEISURE</td>
<td>1</td>
<td>1</td>
<td>YES</td>
</tr>
<tr>
<td>STATIONERY</td>
<td>1</td>
<td>1</td>
<td>YES</td>
</tr>
<tr>
<td>ARTICLE-RLM</td>
<td>2</td>
<td>2</td>
<td>NO</td>
</tr>
</tbody>
</table>

****** TO MAKE THE SPECIFIED REALMS CONSISTENT, THE FOLLOWING LOG FILES ARE ________________3.

NECESSARY:

FROM ALOG SEQ NR 2 TO ALOG SEQ NR 2
OR FROM LOG INTERVAL BEGIN 20040812154053 TO LOG INTERVAL END 20040812154053

****** LOG MODE : AFIM LOGGING ________________________________________________________4.

****** SUPPORTS OF ACTUAL LOG FILE:

DEFAULT SUPPORT :PVS ID = DEFAULT PVS
RESERVE SUPPORT :PVS ID = :H32:
BMEND

SHOW-LOG-INFORMATION statement

***** INFORMATION ABOUT LOG HISTORY : _________________________________________________

<table>
<thead>
<tr>
<th>ALOG SEQ NR!</th>
<th>LOG INTERVAL</th>
<th>AFIM</th>
<th>BACKOUT</th>
<th>LOGGING</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* BEGIN * END ! BFIM ! GAP *

-----

4 !20040812154113!20040812154114! * ! ! ? !
3 !20040812154111!20040812154113! * ! ! !
2 !20040812154053!20040812154111! * ! ! !
1 !20040812154052!20040812154053! * ! ! !

***** ONLINE COPIES BY ARCHIVE ARE ALLOWED _____________________________________________

***** NORMAL END FUNCTION SHOW LOG INFORMATION AT 15:41:32

***** DIAGNOSTIC SUMMARY OF BMEND

NO WARNINGS
NO ERRORS
NO SYSTEM-ERRORS

***** END OF DIAGNOSTIC SUMMARY

***** NR OF DATABASE ACCESSES : 49

***** NORMAL END BMEND (UDS/SQL V2.4 0400 ) 2004-08-12 15:41:32
Start logging for database original (START-LOG)

The START-LOG statement is used to activate AFIM logging for the original database and to optionally define new volumes on which subsequent ALOG files are to be created.

<table>
<thead>
<tr>
<th>START-LOG</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DEFAULT-SUPPORT</strong> = *PUBLIC(...) / UNCHANGED / list-poss(15): *PRIVATE(...)</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

**DEFAULT-SUPPORT = *PUBLIC (...)**

Any new ALOG file to be created will be created on public disk unless an explicit CREATE-FILE command is issued for the new ALOG file.

**CATID = *OWN**

The default catalog ID of the configuration ID is used.

**CATID = <catid>**

The specified catalog ID is used.
DEFAULT-SUPPORT = list-poss(15): *PRIVATE (...)
Any new ALOG file to be created will be created on the specified disk or disks unless an explicit CREATE-FILE command is issued for the new ALOG file. A maximum of 15 disks may be specified.

VOLUME = list-poss(15): <volume>
Specifies the private disks on which ALOG files can be created.

DEVICE = <device>
Defines the device type of the private disks.

DEFAULT-SUPPORT = UNCHANGED
Existing values remain in effect. UNCHANGED is only possible if logging has already been started.

RESERVE-SUPPORT = *PUBLIC (...)
The ALOG file will be created on public disk if no explicit specification for the new ALOG file to be created exists, and if the file cannot be created on the default disk or disks.

CATID = *OWN
The default catalog ID of the configuration ID is used.

CATID = <catid>
The specified catalog ID is used.

RESERVE-SUPPORT = list-poss(15): *PRIVATE (...)
Any new ALOG file to be created will be created on the specified disk if the file cannot be created on the default disk or disks.

VOLUME = list-poss(15): <volume>
Specifies the private disks on which ALOG files can be created.

DEVICE = <device>
Defines the device type of the private disks.

RESERVE-SUPPORT = UNCHANGED
Existing values remain in effect. UNCHANGED is only possible if logging has already been started.

If the DEFAULT and RESERVE operands are assigned the same value, an error occurs. This error is detected on creating the ALOG files, but not during the syntax analysis of the START-LOG statement. It is not possible to switch to an alternative medium.
SPACE = STD
The new ALOG file to be created is assigned a primary allocation value of 192 and a secondary allocation value of 576.

SPACE = *RELATIVE (...)
The specified values are assigned as primary and secondary allocations for the new ALOG file to be created.
(The values specified here must be \( \geq 192 \) and \( \leq 50331645 \) for the primary allocation, and \( \geq 576 \) and \( \leq 32767 \) for the secondary allocation.)

\[
\text{PRIMARY-ALLOCATION} = \langle \text{integer }192..50331645\rangle
\]
Number of PAM pages for the primary allocation.

\[
\text{SECONDARY-ALLOCATION} = \langle \text{integer }576..32767\rangle
\]
Number of PAM pages for the secondary allocation.

SPACE = UNCHANGED
Existing values remain in effect. UNCHANGED is only possible if logging has already been started.

USER-ACCESS = *OWNER-ONLY
Restricts access to the ALOG file to the user ID under which it was created.

USER-ACCESS = *ALL-USERS
Permits the ALOG file to be accessed by other user IDs as well.

RESET-LOG-POOL = NO
The new ALOG file is created with an ALOG sequence number that is obtained by incrementing the ALOG SEQ NR from the Act-Key-0 of the DBDIR by 1.

RESET-LOG-POOL = YES
The ALOG file begins with ALOG SEQ NR = 1.
This operand is typically used to create the log pool of a duplicated database - starting with ALOG SEQ NR = 1.
You can specify up to 15 disks in a PRIVATE operand and up to 15 variants of the PRIVATE operand. However, if the number of disks that are specified in a statement exceeds 15, the last 15 entries apply.

This means that it is only worthwhile specifying multiple variants of the PRIVATE operand if the disks are assigned to different device types.

If an ALOG file cannot be accessed (because it was deleted, for example), a new ALOG file with an ALOG sequence number incremented by 1 is created by the utility routine. If the deleted ALOG file had no relevant information (no deviation from consistency point), the update could be applied in two steps despite the logging gap produced as a result of the deletion (step 1 up to the gap; step 2 after it).

**Example**

1. Activate AFIM logging and the online backup capability. New ALOG files are to be created on different public disks.

```
/ASSIGN-SYSDTA TO-FILE=*SYSCMD
/SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL, VERSION=02.4A00
/START-UDS-BMEND
***** START BMEND (UDS/SQ\L V2.4 0400 ) 2004-08-12 13:39:05
//ALLOCATE-BUFFER-POOL BUFFER-SIZE=*STD
//OPEN-DATABASE DATABASE-NAME=SHIPPING
***** DATABASE ORIGINAL WITHOUT AFIM LOGGING
   FUNCTION ENABLE NOT AVAILABLE
   FUNCTION KILL NOT AVAILABLE
***** CONSISTENT DATABASE DIRECTORY
//START-LOG DEFAULT-SUPPORT=*PUBLIC(CATID=*OWN), -
   RESERVE-SUPPORT=*PUBLIC(CATID=H32), -
   SPACE=STD,RESET-LOG-POOL=NO
***** LOGGING WILL BE ACTIVATED
   FUNCTION ENABLE AVAILABLE FROM NOW ON
//ENABLE-ONLINE-COPY
//END
***** BEGIN FUNCTION START LOGGING AT 13:39:05
***** NORMAL END FUNCTION START LOGGING AT 13:39:06
***** BEGIN FUNCTION ENABLE ONLINE COPY AT 13:39:06
***** ONLINE COPY FOR DATABASE $XXXXXXXX.SHIPPING ALLOWED
***** NORMAL END FUNCTION ENABLE ONLINE COPY AT 13:39:06
***** DIAGNOSTIC SUMMARY OF BMEND
   NO WARNINGS
   NO ERRORS
   NO SYSTEM-ERRORS
```
2. Reset the ALOG SEQ NR to 1 and specify three disks of the same device type on which new ALOG files can be subsequently created.

```
/ASSIGN-SYSDTA TO-FILE=*SYSCMD
/SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL,VERSION=02.4A00
/START-UDS-BMEND
//OPEN-DATABASE DATABASE-NAME=dbname
//START-LOG
  // DEFAULT=*PRIVATE(VOLUME=(G3200A,G3200B,G3200C),-
  // DEVICE=D3468) ,-
  // RESERVE=*PRIVATE(VOLUME=G3400A ,-
  // DEVICE=D3468) ,-
  // RESET-LOG-POOL=YES
//END
***** BEGIN FUNCTION START LOGGING AT timestamp
***** NORMAL END FUNCTION START LOGGING AT timestamp
```

3. Assign three disks of different device types on which new ALOG files can be subsequently created.

```
/ASSIGN-SYSDTA TO-FILE=*SYSCMD
/SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL,VERSION=02.4A00
/START-UDS-BMEND
//OPEN-DATABASE DATABASE-NAME=dbname
//START-LOG DEFAULT=(*PRIVATE(VOLUME=G3030M, ,-
  // DEVICE=D3468) ,-
  // *PRIVATE(VOLUME=(G2065C,G2065D) ,-
  // DEVICE=D5804) ) ,-
  // RESERVE=*PUBLIC
//END
***** BEGIN FUNCTION START LOGGING AT timestamp
***** NORMAL END FUNCTION START LOGGING AT timestamp
```

`timestamp` indicates the current time.
Stop logging for database operations (STOP-LOG)

The STOP-LOG statement deactivates logging. Before logging is stopped, BMEND disables the online-copy facility for all realms if required (i.e. if previously enabled).

```
STOP-LOG
```

This statement has no operands.
Undo a statement (UNDQ)

The UNDO statement cancels the last correctly entered statement (except for UNDO itself). In other words, that statement is not executed.

Each subsequent UNDO statement cancels the preceding statement in the chain (except for the UNDO statement itself).

The UNDO statement does not cancel the ALLOCATE-BUFFER-POOL and END statements.

| UNDO |

This statement has no operands.
Apply AFIMs to a database (UPDATE-DATABASE)

The UPDATE-DATABASE statement can be used to apply AFIMs from ALOG files to realms and thus update them. The required realms, which represent an older status of the database, must be first copied, read in from ARCHIVE backups, or recataloged and made available. Detached realms can also be processed if BMEND is run in parallel with the DBH. Inconsistent realms must be made consistent with the UPDATE-DATABASE statement before they are attached or detached.

```
UPDATE-DATABASE

REALM-NAME  =  *ALL / *ALL-EXCEPT(...) / list-poss(30): <realm-name>
   *ALL-EXCEPT(...)
      NAME  =  list-poss(30): <realm-name>

,DEADLINE  =  STD / BREAK-POINT / <alog-seq-no> / *TIME-STAMP(...)
   *TIME-STAMP(...)
      DATE  =  <date>
      ,TIME  =  <time>

,DELETE  =  NO / YES
```

**REALM-NAME = *ALL**
All realms are updated.

**REALM-NAME = *ALL-EXCEPT(...)**
All realms except for those specified are updated.

```
   NAME  =  list-poss(30): <realm-name>
   Name of the realm that is not to be updated.
```

**REALM-NAME = list-poss(30): <realm-name>**
All specified realms are updated.

**DEADLINE = STD**
The database is updated to the end of the last ALOG file closed. A consistency point recorded in an ALOG file that is not closed yet cannot be reached with DEADLINE = STD.
DEADLINE = BREAK-POINT
All consistent ALOG files, including the last, current, and possibly inconsistent ALOG file are used for the update. This allows an update up until the point at which failure occurred. If the last ALOG file is inconsistent, a warm start of the database is required afterwards. Realms of the shadow database are only updated with closed ALOG files.

DEADLINE = <alog-seq-no>
Sequence number of the ALOG file up to and including which AFIMs are to be applied. Only closed ALOG files are used for the update (leading zeros in the alog-seq-no may be omitted in the specification.)

DEADLINE = *TIME-STAMP(…)
Updates are applied to the database up to and including the last closed ALOG file for which the LOG_INTERVAL_END is less than or equal to the given *TIME-STAMP(…).

DATE = <date>
Date that limits the applied updates.

TIME = <time>
Time that limits the applied updates. If the time is not unique, daylight saving time (summer time) is assumed, and a warning is issued.

DELETE = NO
The applied ALOG files are retained on disk.

DELETE = YES
With the exception of the current file, all applied ALOG files (i.e. the files read in for the update) are automatically deleted, assuming that no error has occurred when executing this statement.

The DBDIR and the DBCOM are treated like any other realm. The realm DBCOM is implicitly addressed by the *ALL and *ALL-EXCEPT options.

If all the realms specified using the REALM-NAME operand are not available, the missing realms are not processed. Furthermore, no deletion of the ALOG file takes place even if requested.

The DEADLINE = BREAK-POINT operand is meaningless for shadow databases. BREAK-POINT is treated as STD in such cases.

The sequence number of the next ALOG file required to reach the desired DEADLINE is saved in a job variable after each individual ALOG file has been applied (see the section on “Supplying job variables” on page 51).
If a logging gap is contained in the sequence of ALOG files to be applied, the update is terminated at that point.
A warning is issued if the DEADLINE (specified as an alog-seq-no or *TIME-STAMP(...)) could not be reached.
A realm other than the DBDIR can only be updated if it was addressed in the update session or matches the DBDIR, which is also being updated.

Statistics and summary report

On completion of the BMEND run, internal counters are evaluated and output for the SUMMARY REPORT:

***** DIAGNOSTIC SUMMARY OF BMEND

{ NO } WARNINGS
{ num } ERRORS
{ num } SYSTEM ERROR

***** END OF DIAGNOSTIC SUMMARY

***** NR OF DATABASE ACCESSES: number
2.2.4 Command sequence to start BMEND

The command sequence described here is based on the assumption that UDS/SQL was installed with IMON (see the section “START commands of the UDS/SQL programs” in chapter 2 of the “Creation and Restructuring” manual).

01 [/CREATE-JV-LINK JV-NAME=JOBVAR,PROTECTION=*STD]
02 [/SET-JV-LINK LINK-NAME=*JVBMEND,JV-NAME=JOBVAR]
03 [/ADD-FI LE-LINK LINKNAME=DATABASE, FILE-NAME=[/catid/][$userid.]dbname.DBDIR]
04 /SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL, VERSION=version
05 /START-UDS-BMEND
06 ///OPEN-DATABASE DATABASE-NAME=dbname [,COPYNAME=*NONE/copyname] [,USER-IDENTIFICATION=*OWN/userid]
07 bmend statements
08 //END

01, 02 Creates a job variable.
03, 06 You must specify one of the two statements.
04 The version of the utility routine is selected.
Specification of the version is generally recommended, since several UDS/SQL versions can be installed in parallel.
05 Alias names for the call are START-UDS-REPAIR and BMEND.

Examples on the use of BMEND can be found in the “Database Operation” manual.
2.3 Supplying job variables

In order to implement automatic database saving and recovery operations, the BMEND utility routine stores information in a job variable. This job variable can be used by other programs or procedures for control purposes.

The job variable is supplied with information if a job variable has been created with LINK-NAME=*JVBMEND.

BMEND does not use the contents of the job variable as input, but simply updates it with relevant values at the end of certain functions.

The process of supplying this job variable with information is internally organized in two parts:

1. initialization with SHOW-LOG-INFORMATION
2. updating with UPDATE-DATABASE

The initialization of the job variable (SHOW-LOG-INFORMATION statement) provides an initial decision support system for the initiation of recovery procedures.

As the user, you must then decide whether the information returned is sufficient. If necessary, you may have to repeat the initialization by modifying the SHOW-LOG-INFORMATION statement with the LOG-FILE operand.

The job variable contains the following information areas:

- status of the processed DBDIR
- status of a log pool segment (common log data)
- status of a log pool segment (AFIM log data)
- sequence numbers to apply further updates

Statement sequence

```
/CREATE-JV JV-NAME=JOBVAR,PROTECTION=*STD
/SET-JV-LINK LINK-NAME=*JVBMEND,JV-NAME=JOBVAR
/ADD-FILE-LINK LINK-NAME=DATABASE,FILE-NAME=SHIPPING.DBDIR
/SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL,VERSION=version,SCOPE=*TASK
/START-UDS-BMEND
//SHOW-LOG-INFORMATION,LOGFILE=STD,OUTPUT=SYSOUT
//END
```

Structure of the job variable

DISPL

( 0) DBDIR_DATA

Describes the status of the accessed DBDIR; deleted by UPDATE

( 0) ALOG_SEQ_NR

ALOG_SEQ_NR from the DBDIR

( 4) CONSISTENCY_DATA

BACK_UP_DATA from the DBDIR; time at which the database was last updated

(18) CONSISTENT

CONSISTENCY / 'Y' or 'N'
C'N': if SYSTEM_BREAK in AK0 or AKn is set
or if AK0 is not equal to AKn

(19) FILLER

Indicates the limits of the examined ALOG file sequence;
the oldest and most recent entry in the history is deleted by UPDATE

(20) COMMON_LOG_DATA

(20) HIGHEST_ALOG_SEQ_NR

Highest sequence number for which the history is stored in the ALOG file

(24) LOWEST_ALOG_SEQ_NR

Lowest sequence number for which the history is stored in the ALOG file

(28) LOG_POOL_PART_END_DATA

YYYYMMDDHHMMS; LOG_INTERVAL_END of the ALOG file assigned in the LOG-FILE operand

(42) LOG_POOL_PART_BEGIN_DATA

YYYYMMDDHHMMS

(56) AFIM_LOG_DATA

Details on most recent log interval with contiguous AFIM logging from the history;
equal to 0 if examined area has no AFIMs; is deleted by UPDATE

(56) UPPER_ALOG_SEQ_NR

Highest sequence number of examined log pool segment with AFIMs

(60) LOWER_ALOG_SEQ_NR

Lowest sequence number of examined log pool segment with AFIMs

(64) UPPER_ALOG_DATA

LOG_INTERVAL_END of ALOG file

(78) LOWER_ALOG_DATA

LOG_INTERVAL_BEGIN of ALOG file
( 92) BACKOUT_LOG_DATA
  Indicates the limits of the most recent BACKOUT
  area without gaps

( 92) UPPER_ALOG_SEQ_NR
  Reserved for future extensions

( 96) LOWER_ALOG_SEQ_NR
  Reserved for future extensions

(100) UPPER_ALOG_DATA
  Reserved for future extensions

(114) LOWER_ALOG_DATA
  Reserved for future extensions

(128) NEXT_SEQ_NR
  Contains the sequence number of the next ALOG
  file to be applied

(128) UPDATE_START_SEQ_NR
  Lowest ALOG number of all examined realms
  initialized by SHOW-LOG;
  is incremented by 1 after reading in all AFIMs of
  an ALOG file;
  equal to 0 if the DEADLINE is reached

(132) RESET_START_SEQ_NR

(136) CHAR_TYPE_LOG_DATA
  Details in character representation

(136) DBNAME
  Name of processed database

(153) COPYNAME
  CHAR (8);
  COPYNAME of processed shadow database

(161) ALOG_SEQ_CHAR
  CHAR (9); ALOG_SEQ_NR from DBDIR

(170) UPDATE_START_SEQ_CHAR
  CHAR (9); UPDATE_START_SEQ_NR

(179) END
Results of initialization

Information from the ALOG BOTTOM PAGE of the assigned ALOG file and from the DBDIR are used for initialization.

The difference between the lowest and highest sequence number of the ALOG files sequence can also be less than 63. This situation is possible with a smaller number of ALOG files or after an inconsistent switch in the ALOG file (old ALOG file no longer accessible).

If an original database is involved, initialization of the job variable returns the following information:

– consistency of the original DBDIR
– time at which the DBDIR was last updated
– highest sequence number of the ALOG file sequence
– the most recent interval with AFIM logging is output (limit values as sequence numbers and with time stamps)

The time of a maximum DEADLINE will have also been stored (LOG END of the most recent AFIM logging interval).

The returned value can be used to read in a suitable recovery log for the UPDATE function.

Since only the last 63 log files are examined, it is not always clear whether the logging period with no gap extends further in the past.

If the ALOG file sequence needs to be examined by going back further in the past, the job variable will need to be reinitialized. When this is done, it is generally advisable to begin with the lowest alog-seq-no of the examined sequence in order to obtain overlapped segments.

If a shadow database was assigned, it is not possible to obtain any information from the DBDIR with regard to which ALOG file sequence numbers were used when the database was last processed. The highest sequence numbers of the used ALOG files can only be determined if the original is assigned. In the case of shadow databases, only the sequence number at the time of saving the DBDIR can be output.

Job variable fields are supplied with information in the same way for a shadow database as when an original is assigned.
Updating with UPDATE

The UPDATE statement results in the deletion of all values of job variables that affect the DBDIR or backout logging.

When an ALOG file has been read in (i.e. applied), the sequence number of the next ALOG file to be applied is entered. When the DEADLINE is reached, a binary zero is entered.

Statement sequence

```
/CREATE-JV JV-NAME=JOBVAR,PROTECTION=*STD
/SET-JV-LINK LINK-NAME=*JVBMEND,JV-NAME=JOBVAR
/ADD-FILE-LINK LINK-NAME=DATABASE,FILE-NAME=SHIPPING.DBDIR
/SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL,VERSION=version,SCOPE=*TASK
/START-UDS-BMEND
//UPDATE-DATABASE REALM-NAME=*ALL,DEADLINE=STD,DELETE=NO
//END
```

3 Checking the consistency of a database with BCHECK

For the database to operate smoothly it is vital that the physical structures of the database be correct. It is, however, not always possible to preclude physical inconsistencies due to system errors. The UDS/SQL utility routine BCHECK allows the user to check the physical structures of the database when problems occur or within the course of periodic data saving and thus uncover potential physical inconsistencies at an early stage. Since BCHECK can pinpoint each error precisely, it is possible to correct a database containing errors and thus avoid further errors.

BCHECK uses the redundancy principle when checking UDS databases. First of all, it checks the local data in a page on the one hand and, on the other, the predefined physical structures and the DBDIR metadata on the basis of the SIA (local consistency). Next, BCHECK checks the consistency of logically associated system data located on different pages, again on the basis of the SIA (global consistency).

It is not possible to check the logical consistency of user data, since no information concerning the reference data or the validity of the user data in the database is available to UDS/SQL.

The following can be checked: the user realms of the database, the PRIVACY-AND-IFQ database in the DBDIR, and the compiler database in the DBCOM.

BCHECK only checks realms on disk.

It is not always necessary to check the entire database: it is also possible to restrict the check range to individual realms, record types, sets or search keys and consistency criteria, thereby saving time. The depth of the check can be likewise restricted by excluding the management structures for the storage of records or the key values of ASC keys, DESC keys or search keys from the check.
3.1 Description of the checking procedure

To optimize the number of accesses to the database, BCHECK checks the database in a single sweep.

BCHECK initially makes local checks in the specified realms on:
– the Act-Key-0 page and Act-Key-N page,
– FPA pages, DBTT anchor pages and DBTT pages,
– the page header of all database pages,
– the formal structure of CALC and table pages,
– the record displacements in the page index entries, and
– the sort sequence of the keys or the record sequence numbers in the table pages

i.e. it checks the internal page structures of the realms.

Subsequently BCHECK checks the global relationships of the specified check objects, i.e. those relationships which transcend page structures. To do this it reads the relevant DBTT entries, SCD entries, table headers, table entries and records from the record types, sets and search keys to be checked and generates what are known as the check records from this information.

3.1.1 Setting the checking mode

For the following error analysis, you specify the mode in which BCHECK further processes the check records. Two checking modes are possible: a summing check and a sort check.

For both modes, BCHECK makes use of the fact that in the various page types (DBTT pages, data pages etc.), information on the check objects is stored in redundant form. For example, from a DBTT page, the page address of the record can be determined for each DB key. If the database is consistent, both DB key and page address are also in the data page in which the record is stored. This means that from the various pages two identical check records are generated for each check object, and, depending on their origin, these will have either a positive or a negative sign.
BCHECK has two procedures for detecting inconsistencies in the database:

1. The counter procedure

   BCHECK sets up three counters:
   - It uses one to count whether precisely the same number of positive and negative check records exist, which is the case if the database is consistent.
   - In the other two, it adds the check records compressed to 6 bytes and the square of these values, respectively.

   This technique, which is based on the theory of error detecting codes, ensures with a high degree of accuracy that an existing error will be detected because the corresponding sums do not match.

   The counter procedure can detect and roughly locate errors in the database but not pinpoint them. This checking method is also called summing check.

2. Sorting procedure

   BCHECK collects all the check records in a file and sorts them so that associated sets of check records must be located next to one another if the database is correct. It then compares each set in turn; if sets do not match, there is an inconsistency at that point in the database.

   This means that BCHECK can use this checking method not only to detect inconsistencies in the database but also to pinpoint them so that the inconsistencies can be rectified. However, because the check records have to be sorted, this procedure is far more time-consuming than the counter procedure.

**Summing check**

Depending on the BCHECK statements, BCHECK uses the counter procedure, the sorting procedure or, as part of a summing check, a combination of the two. Since the sorting procedure in the summing check is only used for a restricted set of check records, the summing check takes very little time and is ideally suited for use during daily data saving activities. If global summation inconsistencies are detected, a sort check would be needed to pinpoint the error location.

**Sort check**

Here BCHECK uses only the sorting procedure. If run unmodified, this check is considerably more time-consuming than the summing check.

You can greatly reduce this time requirement by performing an incremental check. This procedure is described below and can also be used for summing checks.
3.1.2 Defining the scope of checking

It is possible to define both the checking mode and the scope of checking. BCHECK has facilities for overall checking and incremental checking.

Overall checking

BCHECK checks the database in its entirety for consistency.

The following can be checked: the original database or a shadow database.

Overall checking must always be performed if there is no consistent copy of the database or the existing copy has not yet been checked.

It is also required if the database has been restructured or reorganized, and there is either no consistent copy of the changed database or the existing copy has not yet been checked.

Incremental checking

BCHECK only checks the pages that have changed with respect to a shadow database of an earlier database status. BCHECK can determine which page contents have changed by comparing the previous shadow database with the original database or with a more recent shadow database. The older shadow database need not have a DBDIR.

Incremental checking saves a considerable amount of time when compared to overall checking, since in most cases BCHECK does not have to check all pages in the database. The extra time involved in reading every non-empty page twice is counterbalanced by the saving involved in checking only the modified pages and - in SORTING mode - by what is generally a far smaller volume of data to be sorted.

In SUMMING mode, the performance difference between overall checking and incremental checking is not very large. However, there is no general rule concerning the performance of the various checking modes in combination with the scope of checking.

3.1.3 Checking for coherence

BCHECK checks whether realms which have been specified as part of a group actually belong to the same version, i.e. whether they are coherent. To do this, it reads from each realm specified the internal version number entered in the DBDIR and the time at which the last change was made (cf. “Consistency time stamp”) and compares these values with the values entered in the realm itself. Only those realms for which BCHECK establishes consistency with the DBDIR are coherent.

Following utility routine runs with BALTER and BREORG, coherency within an incremental check is not ensured.
3.2 System environment

Effects on database operation

If BCHECK accesses the original database during the check run, this is referred to as an online check run and can be executed in parallel with SHARED-RETRIEVAL database operation.

Check runs in which BCHECK does not access the original database are known as offline check runs and can be executed in parallel with any database operation mode.

More than one check run can be executed in parallel.

Before each BCHECK run, the database to be checked or the shadow database must be assigned with the following command:

```
/ADD-FILE-LINK LINK-NAME=DATABASE,FILE-NAME=dbname.DBDIR[.copy-name]
```

Coherence checking

During each check run, BCHECK performs a coherence check on the databases for which it can access the DBDIR. The following databases are checked for coherence:

- **For overall checking:**
  the original database or the shadow database

- **For incremental checking of original ←→ shadow database:**
  the original database and the shadow database, provided its DBDIR is available

- **For incremental checking of new shadow database ←→ old shadow database:**
  the newer shadow database and the older shadow database, provided its DBDIR is available
The following diagrams show the files needed for the various check runs:

*Overall check of original*

Original of each realm to be checked

Original of DBDIR

Figure 4: System environment for an overall check of original realms
Overall check of shadow database

Realms of the shadow database that are to be checked

DBDIR of the shadow database

Figure 5: System environment for an overall check of the shadow database
Incremental check of original ↔ shadow database

Original of each realm to be checked
Original of the DBDIR
Realms of the shadow database
Possibly, DBDIR of the shadow database (see page 61)

Figure 6: System environment for an incremental check of original ↔ shadow database
Incremental check of new shadow database ↔ old shadow database

Realms of each shadow database that are to be checked

DBDIR for older shadow database or both DBDIRs (see page 61)

Figure 7: System environment for an incremental check of shadow database ↔ shadow database
Required work files

BCHECK requires two work files for a check run which it sets up automatically on public disk under the user identification under which it was started and which it deletes again after the run has terminated normally. The default link names of these files are SCRTCH1 and SORTWK:

SCRTCH1
BCHECK requires this file during each check run to store a page directory.

SORTWK
Requires the SORT used by BCHECK in the case of a sort check and when checking index values in order to collate and sort the check records. See also the manual “SORT (BS2000/OSD)”.

If the two work files are to be created explicitly, they must be assigned the following attributes:

Work-file-1
File link name: SCRTCH1
Access method = SAM; fixed record length

The data population for buffering can be calculated using the following formula:

\[
\text{number-of-pages} \times 16 \text{ Bytes}
\]

\text{number-of-pages}
Number of relevant pages of all realms to be checked, i.e. the sum of the realm sizes in PAM pages minus the act-key-0 pages, FPA pages, and empty pages.

The primary allocation for Work-file-1 should be based on the data population that is to be buffered. There should always be an appropriate secondary allocation in case the storage space proves to be insufficient.
**Work-file-2**

File link name: SORTWK

Access method = PAM

In the case of an overall check, the data population for sorting can be calculated using the following two formulae

- **RSQ check formula:**
  \[ 26 \times \text{number-of-check-records} \text{ Bytes} \]

- **Index value check formula:**
  \[ (\text{max-key-length} + 32) \times \text{number-of-check-records} \text{ Bytes} \]

  \( \text{max-key-length} \)
  Length of the longest key to be checked during the index check (see page 78 and page 80).

You can ascertain the population involved in a SORTING run without an index value check by first performing CHECK SUMMING, calculating twice the total amount of check objects from RECORD/TABLE-OCURRENCES, CHAIN-SET-MEMBERSHIPS, REFERENCES BETWEEN TABLE-OCURRENCES and REFERENCES FROM TABLES TO MEMBER-RECORDS as given by DIAGNOSTIC SUMMARY OF BCHECK and entering this value as \( \text{number-of-check-records} \) in the RSQ check formula.

Similarly, when performing a SORTING run with index value checking, you first calculate twice the value obtained from RECORD/TABLE-OCURRENCES, CHAIN-SET-MEMBERSHIPS and REFERENCES FROM TABLES TO MEMBER-RECORDS and use this value as \( \text{number-of-check-records} \) in the RSQ check formula. You now obtain the population by adding the number of bytes identified in this way to the result of the index value check formula, using twice the value of REFERENCES BETWEEN TABLE-OCURRENCES as \( \text{number-of-check-records} \).

In the case of incremental checks, the population refers to the changes compared to the comparison state.

SORT needs Work-file-2 if the main memory affected by the SORTCORE statement is inadequate. The primary allocation should therefore be based on the data population that is to be sorted. There should always be an appropriate secondary allocation in case it is necessary to extend the storage space.
3.3 Using the results of the summing run in a sort run

Using internal results of summing checks

On detecting global summation inconsistencies BCHECK generates an output file named

UTI.tsn.time-stamp.BCHECK

tsn Four-digit task sequence number
time-stamp Date and time of file generation Format: ddhhmmss

This file holds the internal results from the summing run, which BCHECK can then evaluate
as input information for a subsequent sort run.

In order to have BCHECK evaluate the output file from the summing run, the following
actions must be completed before the sort run:

1. The output file UTI.tsn.time-stamp.BCHECK must be assigned the file link name BCHECK,
2. And GENERATE SORTING must be specified in the CHECK statement for the sort run.

If there is no usable data for a SORTING run in the UTI file, the file will be deleted when
global errors occur.

Evaluating the output log

In a summing run BCHECK writes the following information to SYSLST:

- all messages and, on detecting global summation inconsistencies,
- the command sequence for a sort run coordinated with the summing run (excluding the
  CREATE-FILE and SET-FILE-LINK commands for the two work files SORTWK and
  SCRTCH1).

Editing the SYSLST file

If you assign SYSLST to a file prior to a summing run, you can edit the BCHECK output
using EDT. The edited file with the generated SORTING statements can be used for the sort
run.

Within the summing run, BCHECK identifies the record types, sets or keys in which errors
are present, and the consistency criteria involved. The SORTING statements generated
contain only these objects and the corresponding TYPE clauses. Realm selection is limited
to the minimum necessary.
3.4 Statements for BCHECK

<table>
<thead>
<tr>
<th>Statement</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>SORTCORE</td>
<td>Optional; define size of sort buffer</td>
</tr>
<tr>
<td>CHECK</td>
<td>Select checking mode and define scope of checking</td>
</tr>
<tr>
<td>TYPE</td>
<td>Optional; specify consistency criteria</td>
</tr>
<tr>
<td>SCHEMA NAME</td>
<td>Identify schema</td>
</tr>
<tr>
<td>REALM NAME</td>
<td>Specify realms to be checked</td>
</tr>
<tr>
<td>RECORD NAME</td>
<td>Optional; specify record types to be checked</td>
</tr>
<tr>
<td>SET NAME</td>
<td>Optional; specify sets to be checked</td>
</tr>
<tr>
<td>KEY REF</td>
<td>Optional; specify search keys to be checked</td>
</tr>
<tr>
<td>END</td>
<td>Terminate statement input</td>
</tr>
</tbody>
</table>

Table 8: BCHECK statements

Entering BCHECK statements interactively

The BCHECK statements comprise

– the control statements
  SORTCORE, CHECK, TYPE and SCHEMA

– the object selection statements
  REALM, RECORD, SET and KEY.

You may enter statements within each group in any desired order. You must enter a CHECK statement prior to your first object selection statement. Once you have entered an object you may not enter any more control statements, nor may you correct any control statements entered.

All statements can be corrected and re-entered interactively.
Define size of sort buffer (SORTCORE)

\[ \text{SORTCORE IS } n \]  

\[ n \quad \text{Number of main memory pages for the sort buffer in 4-Kbyte units; the value range for } n \text{ is predefined by the SORT.} \]

Default value: 150

The BS2000 utility routine SORT is used by BCHECK in sort checking to sort the check records (see the “SORT (BS2000/OSD)” manual).

The SORTCORE statement is used to determine the amount of main memory space used for the sort buffer of the SORT utility routine (see the “SORT (BS2000/OSD)” manual). It is optional for sort checking.

The data population for sorting is the same as that on which the size of Work-file-2 is based (see “Work-file-2” on page 67).
Select checking mode and define extent of checking (CHECK)

```
CHECK[ { [GENERATE ]SORTING } ] [ AGAINST COPY NAME IS copy-name ]
```

GENERATE SORTING
Sort check with evaluation of internal results of earlier summing run

SORTING
Sort check

SUMMING
Summing check

AGAINST COPY
Incremental check

`copy-name`
Copy name of the shadow database

Default value:
SUMMING

The CHECK statement is used to define whether BCHECK is to perform a sort check or a summing check and an overall check or an incremental check.

You must enter the CHECK statement among the control statements.

⚠️ All local errors should have been eliminated before a sort check is run, as otherwise they might lead to apparently global errors.
Select consistency criteria (TYPE)

\[
\text{[TYPE IS} \begin{cases} \text{ALL} & \text{[EXCEPT type-no-1][, type-no-2]...]} \\ 0 & \text{[type-no-1][, type-no-2]...} \end{cases}\text{]}\]

**ALL**  
BCHECK checks under all consistency criteria.

**type-no**  
Number of consistency criterion (see below) \(type-no=1\ldots11\)

**ALL EXCEPT type-no-1[type-no-2]...**  
BCHECK checks all consistency criteria other than those listed after EXCEPT.

**0**  
BCHECK checks only locally, i.e. within one page. Related check objects in other pages are not checked.

**type-no-1[type-no-2]...**  
BCHECK checks only under the given consistency criteria.

Default value:  
ALL

BCHECK recognizes the following consistency criteria:

<table>
<thead>
<tr>
<th>type-no</th>
<th>Consistency criterion</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Correct referencing of record or top/first table occurrence by Act-key in DBTT column</td>
</tr>
<tr>
<td>2</td>
<td>Correct chaining of records, including owner record, in a MODE IS CHAIN set</td>
</tr>
<tr>
<td>3</td>
<td>Correct chaining of records, including owner record, in a MODE IS CHAIN LINKED TO PRIOR set</td>
</tr>
<tr>
<td>4</td>
<td>Correct chaining between first and last table occurrences at level 0</td>
</tr>
<tr>
<td>5</td>
<td>Correct chaining of table occurrences between levels</td>
</tr>
<tr>
<td>6</td>
<td>Correct chaining of table occurrences between levels in terms of index values</td>
</tr>
<tr>
<td>7</td>
<td>Correct chaining of table occurrences within one level</td>
</tr>
<tr>
<td>8</td>
<td>Correct referencing of records by level 0 table entries</td>
</tr>
<tr>
<td>9</td>
<td>Correct referencing of records by indirect CALC table entries</td>
</tr>
<tr>
<td>10</td>
<td>Correct chaining in CALC table overflow chain</td>
</tr>
<tr>
<td>11</td>
<td>Correct chaining in duplicates table overflow chain</td>
</tr>
</tbody>
</table>

Table 9: BCHECK consistency criteria
The table below shows which consistency criteria are checked for the various check objects.

<table>
<thead>
<tr>
<th>BCHECK statement</th>
<th>Type</th>
<th>Consistency criterion</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECORD WITH LOCATION CHECK</td>
<td>Not-CALC</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Direct-CALC</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Indirect-CALC</td>
<td>X</td>
</tr>
<tr>
<td>WITHOUT LOCATION ..</td>
<td>All</td>
<td></td>
</tr>
<tr>
<td>SET WITH INDEX CHECK</td>
<td>Table</td>
<td>X</td>
</tr>
<tr>
<td>WITHOUT INDEX CHECK</td>
<td>Table</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>CHAIN</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CHAIN PRIOR</td>
<td></td>
</tr>
<tr>
<td></td>
<td>System set CHAIN</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>System set CH-PR</td>
<td>X</td>
</tr>
<tr>
<td>KEY WITH INDEX CHECK</td>
<td>Table</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Duplicates table</td>
<td>X</td>
</tr>
<tr>
<td>WITHOUT INDEX CHECK</td>
<td>Table</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Duplicates table</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Indirect-CALC</td>
<td></td>
</tr>
</tbody>
</table>

Table 10: Combinations of check objects and consistency criteria

The specification “ALL” is recommended in all cases. Only in this way is it guaranteed that the check will be complete.

BCHECK ignores unavailable DB structures even if the corresponding consistency criteria are specified. Performance is therefore not improved by omitting these criteria.
Identify schema (SCHEMA NAME)

**SCHEMA NAME** IS *schema-name*

*schema-name*

Name of the schema whose SIA is to be used by BCHECK in the checking of the database. The following can be specified for *schema-name*:

- *user schema name* to check user realm
- COMPILER-SCHEMA to check the DBCOM
- PRIVACY-AND-IQF-SCHEMA to check the DBDIR

The SCHEMA statement specifies the SIA from which BCHECK is to fetch the schema information for checking the database. If the SCHEMA statement is not specified, the user database is checked.

When checking the DBDIR, BCHECK does not check the record type SSIA-RECORD on account of its special structure (spanned records).
Specify realms to be checked (REALM NAME)

REALM NAME IS {ALL[ EXCEPT realm-name-1,...]
{realm-name-2,...} }

realm-name
Name of a realm in the database; enter information as follows for user realms:
- realm name defined in the schema DDL per AREA clause
- the database directory: DATABASE-DIRECTORY
- the database compiler realm: DATABASE-COMPILER-REALM

ALL BCHECK checks all non-temporary user realms identified as EXISTENT and SWITCHED-ON in the database catalog.
The realms are checked by realm numbers in ascending order.

ALL EXCEPT realm-name-1,...
Same meaning as for ALL; however, BCHECK excludes the listed realms from the check.

realm-name-2,...
Names of the realms - listed individually - which BCHECK is to check.

If multiple REALM statements are specified between the control statements and the END statement, BCHECK checks in the sequence specified. At least one realm must be specified.

All HASH routines belonging to CALC pages in the specified realms must be entered in the module UDSHASH and must be present in the HASHLIB; otherwise, further checking will be declined.
Specify record types to be checked (RECORD NAME)

The RECORD NAME statement specifies for BCHECK the record types to be checked. The statement may be specified as often as desired in any position after the control statements. BCHECK ignores duplicate entries.

\[
\text{RECORD NAME IS} \begin{cases} 
\text{ALL [ EXCEPT recordname-1,...]} \end{cases} \left[ \begin{cases} \text{WITH} \end{cases} \right] \begin{cases} \text{LOCATION CHECK} \end{cases} 
\end{cases} 
\]

- **recordname**
  - Name of a record type of the specified schema;
  - recordname must be located within the specified realm or realms.

- **ALL**
  - BCHECK checks all record types of the named schema which are contained in the realms to be checked.

- **ALL EXCEPT recordname-1,...**
  - Same meaning as for ALL; however, BCHECK excludes the listed record types from the check.

- **recordname-2,...**
  - Names of the record types - listed individually which BCHECK is to check.

- **WITH LOCATION CHECK**
  - BCHECK also checks all management structures associated with the storage of a record type, i.e. in the DBTT the references to the records.
  - BCHECK checks for record types defined with LOCATION MODE IS CALC:
    - the results of hashing (for primary pages and first overflow pages only)
    - the chaining of overflow pages
    - the internal structure of the CALC tables and
    - the references in the CALC table entries to the records

- **WITHOUT LOCATION CHECK**
  - BCHECK only checks local information on record types, i.e. it checks whether the record addresses in the DBTT are plausible and whether the record lengths match the entries in the SIA, but it does not generate any check records.
If two or more CALC keys with the same key length exist in a realm (indirect CALC keys or CALC-SEARCH keys), BCHECK cannot carry out a global check on them unless all CALC keys in this realm have been specified with this key length. If such a key is selected for checking, BCHECK refuses to perform a global check on it but carries out a local check for the hash area concerned. Since for this local check BCHECK uses all those CALC pages which contain CALC keys of the same length as the CALC keys to be checked, it may be the case that BCHECK will report local inconsistencies in CALC pages which do not belong to any of the CALC keys specified.

In a sort check it is generally not possible to pinpoint inconsistencies in CALC keys. BCHECK is only able to determine which realm they are in, the RSQ, and the key length.

The occurrence of illegal duplicates of a CALC key in various pages of a hash overflow chain cannot be determined by BCHECK.
Specify sets to be checked (SET NAME)

\[
\text{SET NAME IS } \begin{cases} \text{ALL [ EXCEPT setname-1, ...]} & \text{[WITH} \\ \text{setname-2, ...} & \text{WITHOUT]} \end{cases} \text{ INDEX CHECK}]
\]

\textit{setname}

Name of a set of the specified schema;
\textit{setname} must be located within the specified realm or realms.

\textbf{ALL}

BCHECK checks all sets of the named schema and their owner record type or member record type, or tables contained in the realms to be checked.

\textbf{ALL EXCEPT setname-1, ...}

Same meaning as for ALL;
however, BCHECK excludes the listed sets from the check.

\textit{setname-2, ...}

Names of the sets - listed individually - which BCHECK is to check.

\textbf{WITH INDEX CHECK}

Check index values (key value and/or record sequence number or either one of the two, depending on the type of table).
Table checks:
BCHECK checks the index values for correctness of inequality relationships which must exist within the chains of table pages of the same level and between the individual table levels in sort key tables, indexed pointer arrays and indexed lists.
This means that BCHECK performs the following checks:

- For chains of table pages of the same level: whether, in forward chaining (forward pointer), the lowest index value of the successor page is greater than the highest index value of the page in question, and vice-versa for backward chaining (with indexed lists, only for levels > 0)
- For table pages of various levels: whether the index value of a table entry at a level > 0 is always greater than or equal to the greatest index value in the next lowest level of the table to which it points and is less than the smallest index value of the table which follows it (with indexed lists, only for levels > 1)

\textbf{WITHOUT INDEX CHECK}

BCHECK does not check index values

Default value:

\textbf{WITHOUT INDEX CHECK}
With the SET statement the sets to be checked are specified to BCHECK. For sets, BCHECK checks

- the owner/member relationship via the set connection data (SCD)
- the chaining of member records in the case of MODE IS CHAIN,
- in the case of non-indexed and indexed lists, sort key tables, and non-indexed and indexed pointer arrays:
  - the table header,
  - the address chaining of table pages of the same level,
  - in the case of indexed tables: the pointers between the table pages of various levels and
  - the references in the lowest level table entries to the records.

For each set that BCHECK checks, it also automatically checks the associated owner record type and member record type without LOCATION CHECK. It is thus only necessary to name these record types in the RECORD statement if it is desired to extend the check depth to the extent of WITH LOCATION CHECK.

The SET statement may be specified as often as desired in any position after the control statements. BCHECK ignores duplicate entries.

Implicit sets, i.e. search keys at record type level and search keys belonging to a set are only checked by BCHECK if this is explicitly requested by the KEY statement.

For index checking (WITH INDEX CHECK), BCHECK sorts the check records even for a summing check.

Index values are only checked by BCHECK in an overall check.

Index checking does not include checking of sort sequences in chains, checking of non-indexed lists, and comparison of key values in table entries and records.
Specify SEARCH keys to be checked (KEY REF)

\[
\text{[KEY REF IS } \begin{cases} \text{ALL[ EXCEPT keyref-1,...]} \end{cases} \begin{cases} \text{WITH} \end{cases} \begin{cases} \text{INDEX CHECK} \end{cases} \begin{cases} \text{WITHOUT} \end{cases} \text{keyref-2,...} \end{cases}\]
\]

**keyref**  Key number of a search key; this can be taken from the SIA PRINT REPORT (see page 122).

**ALL**  Of the schema named, BCHECK checks all search keys and the relevant record type on which the search key is defined as long as they are contained in the realms to be checked.

**ALL EXCEPT keyref-1,...**  
Same meaning as for ALL however, BCHECK excludes the listed keys from the check.

**keyref-2,...**  
Key numbers of the search keys - listed individually - which BCHECK is to check.

**WITH INDEX CHECK**  
Check index values for indexed search key tables (see page 78).

**WITHOUT INDEX CHECK**  
BCHECK does not check any index values.

Default value: 
**WITHOUT INDEX CHECK**

The KEY statement is used to indicate to BCHECK the keys to be checked. The search keys may be defined on the record type level or the set level as INDEX-SEARCH or CALC-SEARCH keys. BCHECK also checks duplicate tables. The record type on which a search key is defined is automatically checked by BCHECK at the same time, but without LOCATION CHECK.

To check search keys, BCHECK carries out the same checks as for

- **CALC-SEARCH keys:**
  - in a LOCATION CHECK of record types which are defined with LOCATION MODE IS CALC (see page 76);

- **INDEX-SEARCH keys:**
  - in the multi-level tables of a set (see page 78).
In the case of duplicate tables, BCHECK also checks
- the reference to the duplicate header in the table header for plausibility,
- the duplicate table header,
- the index values in the table index entries of a table page for ascending sequence,
- the references to the table entries (DB key lists) from the table index entries for plausibility,
- the chaining of the overflow pages of a duplicates table and
- the record sequence numbers of a table entry for ascending sequence.

The KEY statement may be specified as often as desired in any position after the control statements. BCHECK ignores duplicate entries.

ASC keys or DESC keys cannot be specified in the KEY statement. They are automatically included in the check if the appropriate set is specified in the SET statement.

If the number of an ASC or DESC key is specified in the KEY statement, BCHECK rejects the checking of this key.

For CALC-SEARCH keys having the same key length, the same restriction applies as for indirect CALC keys (see page 76).

The occurrence of illegal duplicates of a CALC-SEARCH key in various pages of a hash overflow chain cannot be determined by BCHECK.

For index checking (WITH INDEX CHECK), BCHECK sorts the check records even for a summing check.

Index values are only checked by BCHECK in an overall check.

Index checking does not include the comparison of key values in table entries and records.
3.5 Command sequence to start BCHECK

The command sequence described here is based on the assumption that UDS/SQL was installed with IMON (see the section “START commands of the UDS/SQL programs” in chapter 2 of the “Creation and Restructuring” manual).

Depending on the check run the following commands are required to start BCHECK:

01  
/CREATE-FILE FILE-NAME=work-file-1 ...
  /ADD-FILE-LINK LINK-NAME=SCRTCH1,FILE-NAME=work-file-1.
      ACCESS-METHOD=*SAM]

02  
/CREATE-FILE FILE-NAME=work-file-2 ...
      ACCESS-METHOD=*UPAM]

03  /ADD-FILE-LINK LINK-NAME=DATABASE.
    FILE-NAME=[:catid:][userid.dbname.DBDIR[.copy-name]

04  
/ADD-FILE-LINK LINK-NAME=BCHECK.
    FILE-NAME=[:catid:]UTI.tsn.time-stamp.BCHECK

05  [/ASSIGN-SYSLST TO=filename]

06  /SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL,VERSION=version

07  /START-UDS-BCHECK

08  bcheck statements

09  END

10  [/ASSIGN-SYSLST TO=*PRIMARY]

03  In this case specifying :catid: is permitted (see the “Database Operation” manual).

04  specified in a sort run to make BCHECK evaluate the summing run output file
UTI.tsn.time-stamp.BCHECK

05  specified in a summing run if the output log is to be used in a subsequent sort run.

06  The version of the utility routine is selected.
Specification of the version is generally recommended, since several UDS/SQL
versions can be installed in parallel.

07  The UDS/SQL utility routine can also be started with the alias BCHECK.
3.6 BCHECK examples

The database CUSTOMER (see page 19) appears as shown below after the restructuring operation (see the Creation and Restructuring manual):

![Diagram of CUSTOMER database]

**Figure 8: Database CUSTOMER after restructuring**

After a short period of operation with the restructured database CUSTOMER, it should be checked for consistency using BCHECK. The following states of the database exist:

- shadow database directly after restructuring (*copyname.AFTRESTR*)
- original database (earlier state than AFTRESTR)
Example 1

Complete summing check of the shadow database after restructuring (AFTRESTR):

```
/ADD-FILE-LINK LINK-NAME=DATABASE,FILE-NAME=CUSTOMER.DBDIR.AFTRESTR
/SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL, VERSION=02.4A00
/START=UDS-BCHECK
***** START BCHECK (UDS/SOL V2.4 0400 ) 2004-08-12 15:40:04
CHECK SUMMING
SCHEMA CUSTOMER-LIST
REALM ALL
SET ALL WITH INDEX CHECK
KEY ALL WITH INDEX CHECK
END
***** FOR INDEX CHECKS A SORTING MUST BE PERFORMED EVEN FOR THE SUMMING
PROCEDURE.

***** THE INPUT IS CORRECT, THE CHECK RUN IS STARTING NOW.
***** ANALYSING NEW STATE OF REALM CUSTOMER-RLM FROM FILE
 :H32:$XXXXXXXX.CUSTOMER.CUSTOMER-RLM.AFTRESTR
 71 NON-EMPTY BLOCKS HAVE BEEN ANALYSED.
***** ANALYSING NEW STATE OF REALM FINANCE-RLM FROM FILE
 :H32:$XXXXXXXX.CUSTOMER.FINANCE-RLM.AFTRESTR
 5 NON-EMPTY BLOCKS HAVE BEEN ANALYSED.

***** DIAGNOSTIC SUMMARY OF BCHECK

   NO WARNINGS
   NO ERRORS
   NO SYSTEM-ERRORS

9927 LOCAL CHECKS HAVE BEEN DONE
 78 RECORD/TABLE-OCCURRENCES HAVE BEEN CHECKED AGAINST DBTT
 68 CHAIN-SET-MEMBERSHIPS HAVE BEEN CHECKED
  4 REFERENCES BETWEEN TABLE-OCCURRENCES HAVE BEEN CHECKED
 72 REFERENCES FROM TABLES TO MEMBER-RECORDS HAVE BEEN CHECKED

   NO EASY LOCAL CONSISTENCY ERRORS
   NO FATAL LOCAL CONSISTENCY ERRORS

   NO GLOBAL CONSISTENCY ERRORS
   NO GLOBAL INDEX-CHECK HAS BEEN DONE

***** END OF DIAGNOSTIC SUMMARY
***** NR OF DATABASE ACCESSES : 32
***** NORMAL END BCHECK (UDS/SOL V2.4 0400 ) 2004-08-12 15:40:05
```
Example 2

The original database is checked against the shadow database that was checked by BCHECK before restructuring and found to be error-free (AFTRESTR). While this incremental summing check is performed, BCHECK also checks both databases for coherence:

```
/ADD-FILE-LINK LINK-NAME=DATABASE,FILE-NAME=CUSTOMER.DBDIR
/SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL, VERSION=02.4A00
/START-UDS-BCHECK
***** START BCHECK (UDS/SQ L V2.4 0400 ) 2004-08-12 15:48:55
CHECK SUMMING AGAINST COPY NAME AFTRESTR
SCHEMA CUSTOMER-LIST
REALM ALL
RECORD ALL
SET ALL
KEY ALL
END
```

```
***** THE INPUT IS CORRECT, THE CHECK RUN IS STARTING NOW.
***** ANALYSING NEW STATE OF REALM CUSTOMER-RLM FROM FILE
 :H32:$XXXXXXXX.CUSTOMER.CUSTOMER-RLM
 AGAINST OLD STATE FROM FILE :H32:$XXXXXXXX.CUSTOMER.CUSTOMER-RLM.AFTRESTR
 48 CHANGED BLOCKS HAVE BEEN ANALYSED.
***** ANALYSING NEW STATE OF REALM FINANCE-RLM FROM FILE
 :H32:$XXXXXXXX.CUSTOMER.FINANCE-RLM
 AGAINST OLD STATE FROM FILE :H32:$XXXXXXXX.CUSTOMER.FINANCE=RLM.AFTRESTR
  1 CHANGED BLOCKS HAVE BEEN ANALYSED.

+++++ GLOBAL CONSISTENCY ERRORS IN TYPE-NR : 1.
+++++ GLOBAL CONSISTENCY ERRORS IN REF-NR  : 2.

***** DIAGNOSTIC SUMMARY OF BCHECK
 .
 .
 .

NO WARNINGS
NO ERRORS
NO SYSTEM-ERRORS

11351 LOCAL CHECKS HAVE BEEN DONE
198 RECORD/TABLE-OCCURRENCES HAVE BEEN CHECKED AGAINST DBTT
136 CHAIN-SET-MEMBERSHIPS HAVE BEEN CHECKED
 54 REFERENCES BETWEEN TABLE-OCCURRENCES HAVE BEEN CHECKED
 144 REFERENCES FROM TABLES TO MEMBER-RECORDS HAVE BEEN CHECKED
```
Examples

BCHECK

NO EASY LOCAL CONSISTENCY ERRORS
NO FATAL LOCAL CONSISTENCY ERRORS

++++++ GLOBAL CONSISTENCY ERRORS. DO A SORTING CHECK
NO GLOBAL INDEX-CHECK HAS BEEN DONE

***** END OF DIAGNOSTIC SUMMARY
***** NR OF DATABASE ACCESSES : 71
***** NORMAL END BCHECK (UDS/SQLE V2.4 0400 ) 2004-08-12 15:48:56

Extracts from the SYSLST log:

/REMARK START GENERATED BCHECK-STATEMENTS
/ADD-FILE-LINK LINK-NAME=BCHECK, FILE-NAME=$XXXXXXXX.UTI.0CZS.12154855.BCHECK
/ADD-FILE-LINK LINK-NAME=DATABASE, FILE-NAME=$XXXXXXXX.CUSTOMER.DBDIR
/SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL,VERSION=XX.XXXX
/START-UDS-BCHECK
CHECK GENERATE SORTING AGAINST COPY AFTRESTR
TYPE IS 1
SCHEMA NAME IS CUSTOMER-LIST
REALM NAME IS CUSTOMER-RLM
REALM NAME IS FINANCE-RLM
RECORD NAME IS CUSTOMER WITH LOCATION CHECK
SET NAME IS APPROVED-CREDITS WITHOUT INDEX CHECK
KEY REF IS 2 WITHOUT INDEX CHECK
END

Since errors were detected in the summing check, a sort check is performed to localize them. The command sequence for the limited sort check appears after REMARK in the SYSLST output of the summing check.
Example 3

A limited sort check is performed to localize the errors more precisely. This is done by using the statements listed after REMARK.. in the log of the summing check.

/ADD-FILE-LINK BCHECK, $XXXXX.UTI.0CZS.12154855.BCHECK
/ADD-FILE-LINK LINK-NAME=DATABASE,FILE-NAME=CUSTOMER.DBDIR
/SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL, VERSION=02.4A00
/START-UDS-BCHECK

***** START BCHECK  (UDS/SQI V2,4  0400 )  2004-08-12  15:49:59
CHECK GENERATE SORTING AGAINST COPY AFTRESTR
TYPE IS  1
SCHEMA NAME IS CUSTOMER-LIST
REALM NAME IS CUSTOMER-RLM
REALM NAME IS FINANCE-RLM
RECORD NAME IS CUSTOMER                        WITH    LOCATION CHECK
SET  NAME IS APPROVED-CREDITS                  WITHOUT INDEX CHECK
KEY REF IS  2                                WITHOUT INDEX CHECK
END

***** THE INPUT IS CORRECT, THE CHECK RUN IS STARTING NOW.
***** ANALYSING NEW STATE OF REALM CUSTOMER-RLM FROM FILE
:H32:$XXXXXXXX.CUSTOMER.CUSTOMER-RLM
AGAINST OLD STATE FROM FILE :H32:$XXXXXXXX.CUSTOMER.CUSTOMER-RLM.AFTRESTR
44 CHANGED BLOCKS HAVE BEEN ANALYSED.
***** ANALYSING NEW STATE OF REALM FINANCE-RLM FROM FILE
:H32:$XXXXXXXX.CUSTOMER.FINANCE-RLM
AGAINST OLD STATE FROM FILE :H32:$XXXXXXXX.CUSTOMER.FINANCE-RLM.AFTRESTR
1 CHANGED BLOCKS HAVE BEEN ANALYSED.

***** DIAGNOSTIC SUMMARY OF BCHECK

NO WARNINGS
NO ERRORS
NO SYSTEM-ERRORS

7499 LOCAL CHECKS HAVE BEEN DONE
72 RECORD/TABLE-OCURRENCES HAVE BEEN CHECKED AGAINST DBTT
0 CHAIN-SET-MEMBERSHIPS HAVE BEEN CHECKED
0 REFERENCES BETWEEN TABLE-OCURRENCES HAVE BEEN CHECKED
0 REFERENCES FROM TABLES TO MEMBER-RECORDS HAVE BEEN CHECKED

NO EASY LOCAL CONSISTENCY ERRORS
NO FATAL LOCAL CONSISTENCY ERRORS
Examples

+++++ 1 GLOBAL CONSISTENCY ERRORS

***** END OF DIAGNOSTIC SUMMARY
***** NR OF DATABASE ACCESSES : 71
***** NORMAL END  BCHECK  (UDS/SQL V2.4 0400 )  2004-08-12  15:49:59

Extracts from the SYSLST log:

***** THE FOLLOWING CHANGE OF DATABASE IS INCONSISTENT
CHECK-ELEMENT : 0002 00000024 0002 FFFF 030000CB 01 00 0000 00000000
FOR OWNER-DBKEY 0002-000000000024 DBTT-COL 2
     A TABLE-OCCURRENCE AT TABLE-LEVEL MAX. WAS ADDED TO PAGE 030000CB
     WITH NO CORRESPONDING CHANGE OF THE REFERENCE IN THE DBTT.

.
3.7 Messages

The messages output by BCHECK can be divided into four groups:

- **Warnings**
  
  Warnings indicate circumstances that do not impair normal execution of BCHECK. However, it is possible that not all of the objects specified by the user are checked. Job switch 30 is set.

- **Error messages (errors, system errors)**
  
  Error messages indicate circumstances that impair or prevent normal execution of BCHECK or result in BCHECK aborting before completion. Job switch 31 is set.

- **Execution messages**
  
  Execution messages provide information on BCHECK execution and are output to SYSOUT and SYSLST.

- **Inconsistency messages**
  
  Inconsistency messages indicate inconsistencies in the examined database. They are output to SYSLST. The job switches 26 (EASY LOCAL), 27 (FATAL LOCAL) and/or 28 (GLOBAL) are set.

3.7.1 Warnings

Following input, BCHECK performs an analysis comparing the objects specified with the objects of the schema.

If this analysis shows that not all of the realms needed for the checking requested have been specified, and only partial checking is possible, a corresponding warning is issued.

Conditions that cause warnings to be issued are, for example:

- An indirect CALC area is to be checked and not all CALC areas of the realm with keys with the same key length have been specified.
- A set or key is to be checked and not all realms containing owner or member records or tables have been specified.
- A record type is to be checked and not all realms with records of this type have been specified.
### 3.7.2 Error messages

If the analysis of the specified objects, described above under warnings, indicates that specified objects cannot be checked, corresponding error messages are issued and the BCHECK run is aborted.

Conditions that cause error messages to be issued are, for example:

- No realm has been specified.
- A set or key is to be checked and the realms for owner or member records or tables have not been specified.
- A record type, set or key is to be checked and the DBTT realm has not been specified.

These and other error messages that do not result from user input are explained in the “Messages“ manual. The error messages also cover any system errors that may occur.

### 3.7.3 Execution messages

**Messages relating to execution during the analysis phase**

**CHECK OF INDICES IS IMPOSSIBLE WITHOUT TOTAL CHECK OF THE DATABASE: THE INDEX CLAUSE IS IGNORED.**

**Meaning**
During incremental checking, consistency criterion 6 cannot be checked (correct chaining of table pages between levels with respect to their index values).

**FOR INDEX CHECKS A SORTING MUST BE PERFORMED EVEN FOR THE SUMMING PROCEDURE.**

**Meaning**
In an index check the check records must be sorted, even for a summing check. During input analysis it is determined that consistency criterion 6 is to be checked (correct chaining of table pages between levels with respect to their index values).

******* THE INPUT IS CORRECT, THE CHECK RUN IS STARTING NOW.**

**Meaning**
Input analysis did not generate any error messages. Actual consistency checking is beginning.
Messages relating to the realm currently being processed

***** ANALYSING NEW STATE OF REALM realm-name FROM FILE file-name. number NON EMPTY BLOCKS HAVE BEEN ANALYSED.

Meaning
Total checking, with statistical note

***** ANALYSING NEW STATE OF REALM realm-name FROM FILE file-name AGAINST OLD STATE FROM FILE file-name. number CHANGED BLOCKS HAVE BEEN ANALYSED.

Meaning
Incremental check ORIG/COPY ←→ COPY, with statistical note

Messages relating to consistency checking in summing checks

+++++ GLOBAL CONSISTENCY ERRORS IN \{\text{TYPE-NR}\} \{\text{type-no}\}, \{\text{REF-NR}\} \{\text{ref-no}\}.

- SYSTEM-ANCHOR-DBKEY = 0000-rsq
- KEY-LENGTH = key-length
- DBTT-COL-NR = dbtt-column-no
- AREA-REF = area-ref

\text{type-no} \quad \text{Designates the consistency criteria for which BCHECK has detected inconsistencies (see page 72)}

\text{ref-no} \quad \text{Designates the record types that are faulty with respect to consistency criteria 1-8}

\begin{align*}
\text{type-no} = 1 & \quad \text{Record type for the faulty DBTT} \\
\text{type-no} = 2-8 & \quad \text{Owner record type for the corresponding set}
\end{align*}

\text{0000-rsq} \quad \text{Output in conjunction with ref-no = 0; designates the DB key of an anchor record.}

\text{key-length} \quad \text{Output with consistency criteria 9 and 10; designates the length of the CALC key.}

\text{dbtt-column-no} \quad \text{Output with consistency criterion 11; designates the column number in the DBTT}

\text{area-ref} \quad \text{Output with consistency criteria 9-11}

\begin{align*}
\text{type-no} = 9,10 & \quad \text{Realm in which the indirect CALC table is located.} \\
\text{type-no} = 11 & \quad \text{Realm in which the duplicates table is located.}
\end{align*}
SUMMARY report

***** DIAGNOSTIC SUMMARY OF DATABASE-CHECK

{NO/ number} WARNINGS refers to the warnings

{NO/ number} ERRORS and

{NO/ number} SYSTEM-ERRORS refers to the error messages

If a consistency check was possible, additional statistical messages are issued:

number RECORD/TABLE ...
Number of references checked from the actual key in a DBTT column to the record or to the highest table page or first table page (consistency criterion 1; see page 72)

number CHAIN-SET-...
Number of records checked
– in a set MODE IS CHAIN including the owner record (consistency criterion 2)
– in a set MODE IS CHAIN LINKED TO PRIOR including the owner record (consistency criterion 3)

number REFERENCES BETWEEN ...
Number of references checked between
– the first table page and the last table page at level 0 (consistency criterion 4)
– the table pages between levels (consistency criterion 5)
– the table pages between levels with respect to their index values (consistency criterion 6)
– the table pages within one level (consistency criterion 7)
– a CALC table overflow chain (consistency criterion 10)
– a duplicates table overflow chain (consistency criterion 11)

number REFERENCES FROM ...
Number of references checked from
– table entries at level 0 to the records (consistency criterion 8)
– the indirect table entries to the records (consistency criterion 9)

number LOCAL CHECKS HAVE BEEN DONE ...
Total number of local checks performed in all the realms to be checked.
• **Local check:**

\[
\begin{align*}
\text{NO} & \quad EASY
\end{align*}
\]

LOCAL CONSISTENCY ERRORS.

Number of minor local inconsistencies or fatal local inconsistencies which BCHECK has discovered in the database during the check run

**EASY LOCAL CONSISTENCY ERRORS**

Minor inconsistency, i.e. local error that does not impede database operation.

**FATAL LOCAL CONSISTENCY ERRORS**

Fatal inconsistency, i.e. local error that must be rectified since it impairs or prevents correct database operation.

• **Summing check:**

\[
\begin{align*}
\text{NO GLOBAL CHECK HAS BEEN DONE} \\
\text{NO GLOBAL CONSISTENCY ERRORS} \\
\text{GLOBAL CONSISTENCY ERRORS. DO A SORTING CHECK}
\end{align*}
\]

These messages refer to consistency criteria 1-5 and 7-11; see page 72.

\[
\begin{align*}
\text{NO} & \quad \text{GLOBAL INDEX-CHECK HAS BEEN DONE} \\
\text{NO} & \quad \text{GLOBAL CONSISTENCY ERRORS IN INDEX-CHECK}
\end{align*}
\]

These messages refer only to consistency criterion 6 (correct chaining of table pages between levels with respect to their index values).

• **Sort check:**

\[
\begin{align*}
\text{NO} & \quad \text{GLOBAL CHECK HAS BEEN DONE} \\
\text{NO} & \quad \text{GLOBAL CONSISTENCY ERRORS}
\end{align*}
\]

Number of global inconsistencies with respect to consistency criteria 1-11; see also page 72.

*****END OF DIAGNOSTIC SUMMARY

**Meaning**

End of the SUMMARY report
3.7.4 Inconsistency messages

If BCHECK has detected local or global inconsistencies, it reports these on SYSLST; however, global inconsistencies are only reported during a sort check.

3.7.4.1 Global inconsistencies (without index check)

Error messages for global inconsistencies which are not related to index checking, start with the header line

THE FOLLOWING CHANGE OF DATABASE IS INCONSISTENT

If BCHECK discovers additional entries relevant to a consistent global relationship (e.g. two record entries with the same DB key, with the DBTT entry referring to one of the two), BCHECK lists the consistent relationship after the inconsistency message. The header line:

THE FOLLOWING DATABASE CHANGES MAY BE CONNECTED WITH THE INCONSISTENCY

is followed in this case by the relevant messages without the additional information:

WITH NO CORRESPONDING CHANGE ..... 

In the following, all error messages for global inconsistencies - except for those related to index checking - are explained in alphabetical order. Note that messages containing the text ..... DELETED FROM ..... are only output during incremental checking. These messages always relate to the older shadow database; however, the explanations given for them refer to the original (or the newer shadow database) and are only relevant if the older shadow database is consistent. Should the older shadow database not be consistent, the explanation for the corresponding message with ADDED TO applies to the inconsistency in this shadow database.

A RECORD-OCCURRENCE WITH RSQ rsq WHICH HAS AN (INDIRECT) CALC-(SEARCH-)KEY WITH KEYLENGTH key-length WAS ADDED TO THE PAGE actkey WITH NO CORRESPONDING CHANGE IN THE CALC-TABLE.

Meaning  
In the CALC table of the indirect CALC key or CALC-SEARCH key with key length key-length, the reference to the record with record sequence number rsq stored in page actkey is missing.

A RECORD-OCCURRENCE WITH RSQ rsq WHICH HAS AN (INDIRECT) CALC-(SEARCH-)KEY WITH KEYLENGTH key-length WAS DELETED FROM THE PAGE actkey WITH NO CORRESPONDING CHANGE IN THE CALC-TABLE.

Meaning  
In the CALC table of the indirect CALC key or CALC-SEARCH key with key length key-length, the entry with record sequence number rsq refers to a non-existent record.
FOR AN (INDIRECT) CALC-(SEARCH-)KEY WITH KEYLENGTH key-length AN ENTRY WITH RSQ rsq WAS ADDED TO A CALC-TABLE AT PAGE actkey OF THE DATABASE WITH NO CORRESPONDING CHANGE IN THE RELATED RECORD-OCCURRENCE.

**Meaning**
In the CALC table of the indirect CALC key or CALC-SEARCH key with key length key-length, the entry in page actkey with record sequence number rsq refers to a non-existent record.

FOR AN (INDIRECT) CALC-(SEARCH-)KEY WITH KEYLENGTH key-length AN ENTRY WITH RSQ rsq WAS DELETED FROM A CALC-TABLE AT PAGE actkey OF THE DATABASE WITH NO CORRESPONDING CHANGE IN THE RELATED RECORD-OCCURRENCE.

**Meaning**
In the CALC table of the indirect CALC key or CALC-SEARCH key with key length key-length, the reference to the stored record with record sequence number rsq is missing.

FOR DBKEY dbkey AN ENTRY WAS ADDED TO DBTT-COL-0 POINTING TO PAGE actkey WITH NO CORRESPONDING CHANGE OF LOCATION OF THE RECORD-OCCURRENCE.

**Meaning**
For record dbkey there exists a DBTT entry which refers to page actkey. The record is not contained in this page.

FOR DBKEY dbkey AN ENTRY WAS DELETED FROM DBTT-COL-0 POINTING TO PAGE actkey WITH NO CORRESPONDING CHANGE OF LOCATION OF THE RECORD-OCCURRENCE.

**Meaning**
The DBTT entry for record dbkey in page actkey is incorrect.

FOR DBKEY dbkey THE RECORD-OCCURRENCE WAS ADDED TO PAGE actkey WITH NO CORRESPONDING CHANGE OF A REFERENCE FROM DBTT-COL-0.

**Meaning**
The DBTT entry for record dbkey in page actkey is incorrect.

FOR DBKEY dbkey THE RECORD-OCCURRENCE WAS DELETED FROM PAGE actkey WITH NO CORRESPONDING CHANGE OF A REFERENCE FROM THE DBTT-COL-0.

**Meaning**
For record dbkey there exists a DBTT entry which refers to page actkey. The record is not contained in this page.

FOR OWNER-DBKEY dbkey DBTT-COL col-no A LAST-TABLE-OCCURRENCE WAS ADDED TO PAGE actkey WITH NO CORRESPONDING CHANGE OF THE REFERENCE IN THE TOP-TABLE.

**Meaning**
In the indexed table which is uniquely identified by owner record dbkey and DBTT column number col-no, the reference from the table page of the highest level corresponding to the last table page of the lowest level in page actkey is missing.
Inconsistency messages

FOR OWNER-DBKEY dbkey DBTT-COL col-no a LAST-TABLE-OCCURRENCE was deleted from page actkey with no corresponding change of the reference in the top-table.

Meaning
In the indexed table which is uniquely identified by owner record dbkey and DBTT column number col-no, page actkey, which is referenced in the table header of the highest level, does not contain the last table page of the lowest level.

FOR OWNER-DBKEY dbkey DBTT-COL col-no an entry referring to a table-occurrence at level levno was added to the {table at next higher level|DBTT|system-record} pointing to a page actkey-1 (with the reference coming from page actkey-2) with no corresponding change of location of the table-occurrence pointed at.

Meaning
In the indexed table which is uniquely identified by owner record dbkey and DBTT column number col-no, a table page of level levno, which is referenced from the next higher level (table entry in page actkey-2), or from the DBTT or from the anchor record, is missing from page actkey-1.

FOR OWNER-DBKEY dbkey DBTT-COL col-no an entry referring to a table-occurrence at level levno was deleted from the {table at next higher level|DBTT|system-record} pointing to a page actkey-1 (with the reference coming from page actkey-2) with no corresponding change of location of the table-occurrence pointed at.

Meaning
In the indexed table which is uniquely identified by owner record dbkey and DBTT column number col-no, the reference from the next higher level (table entry in page actkey-2), or from the DBTT or from the anchor record to a table page with level levno in page actkey-1 is missing.

FOR OWNER-DBKEY dbkey-1 DBTT-COL col-no an entry was added to a bottom-table pointing to member-record with dbkey dbkey-2 (with the table being located in page actkey) with no corresponding change of the record-occurrence.

Meaning
In the search key or sort key table which is uniquely identified by owner record dbkey-1 and DBTT column number col-no, there exists for record dbkey-2 a table entry at level 0 (stored in page actkey), although this record is not a member of the corresponding set.

FOR OWNER-DBKEY dbkey-1 DBTT-COL col-no an entry was deleted from a bottom-table pointing to a member-record with dbkey-2 (with the table being located in page actkey) with no corresponding change of the record-occurrence.

Meaning
In the search key or sort key table which is uniquely identified by owner record dbkey-1 and DBTT column number col-no, the table entry at level 0 (stored in the page actkey) corresponding to member record dbkey-2 is missing.
FOR OWNER-DBKEY dbkey DBTT-COL col-no A TABLE-OCCURRENCE AT TABLE-LEVEL levno WAS ADDED TO PAGE actkey-1 POINTING TO NEXT-TABLE IN PAGE actkey-2 WITH NO CORRESPONDING CHANGE OF A BACKWARD-POINTER IN THE TABLE-CHAIN.

Meaning
In the table which is uniquely identified by owner record dbkey and DBTT column number col-no, the backward pointer from page actkey-2 corresponding to the forward pointer in page actkey-1 on the same level levno is missing.

FOR OWNER-DBKEY dbkey DBTT-COL col-no A TABLE-OCCURRENCE AT TABLE-LEVEL {levno|MAX} WAS ADDED TO PAGE actkey-1 (POINTING TO NEXT HIGHER TABLE IN PAGE actkey-2) WITH NO CORRESPONDING CHANGE OF THE REFERENCE IN THE {TABLE AT NEXT HIGHER LEVEL|DBTT|SYSTEM-RECORD}.

Meaning
In an indexed table which is uniquely identified by owner record dbkey and DBTT column number col-no, the reference from the next higher level (table entry in page actkey-2), or from the DBTT or from the anchor record to the table page with level levno in page actkey-1 is missing. For table pages of the highest level, MAX replaces levno.

FOR OWNER-DBKEY dbkey DBTT-COL col-no A TABLE-OCCURRENCE AT TABLE-LEVEL levno WAS ADDED TO PAGE actkey-1 POINTING TO PRIOR-TABLE IN PAGE actkey-2 WITH NO CORRESPONDING CHANGE OF A FORWARD-POINTER IN THE TABLE-CHAIN.

Meaning
In the table which is uniquely identified by owner record dbkey and DBTT column number col-no, the forward pointer from page actkey-2 corresponding to the backward pointer in page actkey-1 on the same level levno is missing.

FOR OWNER-DBKEY dbkey DBTT-COL col-no A TABLE-OCCURRENCE AT TABLE-LEVEL levno WAS DELETED FROM PAGE actkey-1 POINTING TO NEXT-TABLE IN PAGE actkey-2 WITH NO CORRESPONDING CHANGE OF A BACKWARD-POINTER IN THE TABLE-CHAIN.

Meaning
In the table which is uniquely identified by owner record dbkey and DBTT column number col-no, the forward pointer from page actkey-1 corresponding to the backward pointer in page actkey-1 on the same level levno is missing.

FOR OWNER-DBKEY dbkey DBTT-COL col-no A TABLE-OCCURRENCE AT TABLE-LEVEL {levno|MAX} WAS DELETED FROM PAGE actkey-1 (POINTING TO NEXT HIGHER TABLE IN PAGE actkey-2) WITH NO CORRESPONDING CHANGE OF THE REFERENCE IN THE {TABLE AT NEXT HIGHER LEVEL|DBTT|SYSTEM-RECORD}.

Meaning
In an indexed table which is uniquely identified by owner record dbkey and DBTT column number col-no, a table page of level levno in page actkey-1, which is referenced from the next higher level (table entry in page actkey-2), or from the DBTT or from the anchor record, is missing. For table pages of the highest level, MAX replaces levno.
Inconsistency messages

FOR OWNER-DBKEY dbkey DBTT-COL col-no A TABLE-occurrence at table-level levno was deleted from page actkey-1 pointing to prior-table in page actkey-2 with no corresponding change of a forward-pointer in the table-chain.

Meaning
In the table which is uniquely identified by owner record dbkey and DBTT column number col-no, the backward pointer in page actkey-1 corresponding to the forward pointer in page actkey-2 on the same level levno is missing.

FOR OWNER-DBKEY dbkey-1 DBTT-COL col-no the membership-indicator was added to the record with member-DBKEY dbkey-2 (located in page actkey) with no corresponding change of a pointer in some bottom-table.

Meaning
In the search key or sort key table which is uniquely identified by owner record dbkey-1 and DBTT column number col-no, the table entry at level 0 corresponding to member record dbkey-2 (stored in page actkey) is missing.

FOR OWNER-DBKEY dbkey-1 DBTT-COL col-no the membership-indicator was deleted from the record with member-DBKEY dbkey-2 (located in page actkey) with no corresponding change of a pointer in some bottom-table.

Meaning
In the search key or sort key table which is uniquely defined by owner record dbkey-1 and DBTT column number col-no, there exists for record dbkey-2 a table entry at level 0, although this record is not a member of the corresponding set.

FOR OWNER-DBKEY dbkey DBTT-COL col-no the reference to the last-table was added to the top-table pointing to page actkey with no corresponding change of location of the last-table-occurrence.

Meaning
In the indexed table which is uniquely identified by owner record dbkey and DBTT column number col-no, page actkey which is referenced in the table header of the highest level table does not contain the last table page of the lowest level.

FOR OWNER-DBKEY dbkey DBTT-COL col-no the reference to the last-table was deleted from the top-table pointing to page actkey with no corresponding change of location of the last-table-occurrence.

Meaning
In the indexed table which is uniquely identified by owner record dbkey and DBTT column number col-no, the reference from the table page of the highest level corresponding to the last table page of the lowest level in page actkey is missing.
FOR OWNER-DBKEY dbkey-1 SET-REF setref A FORWARD-POINTER WAS ADDED TO THE OWNER'S SET-CHAIN POINTING TO MEMBER-DBKEY dbkey-2 (FROM DBKEY dbkey-3) WITH NO CORRESPONDING CHANGE OF THE MEMBER-RECORD POINTED AT.

Meaning
In set setref, there exists in the set occurrence of owner dbkey-1 a reference to record dbkey-2 which is not, however, a member of the chain.
For backward chaining, member record dbkey-3 which contains the reference is also given.

FOR OWNER-DBKEY dbkey-1 SET-REF setref A FORWARD-POINTER WAS DELETED FROM THE OWNER'S SET-CHAIN POINTING TO MEMBER-DBKEY dbkey-2 (FROM DBKEY dbkey-3) WITH NO CORRESPONDING CHANGE OF THE MEMBER-RECORD POINTED AT.

Meaning
In set setref, no reference to record dbkey-2 exists in the set occurrence of the owner.
For backward chaining, member record dbkey-3 which should have contained the reference is also given.

FOR OWNER-DBKEY dbkey-1 SET-REF setref A RECORD-OCCURRENCE WAS ADDED TO THE OWNER'S SET-CHAIN WITH MEMBER-DBKEY dbkey-2 (POINTING TO PRIOR DBKEY dbkey-3) WITH NO CORRESPONDING CHANGE OF A FORWARD-POINTER IN THE SET-CHAIN.

Meaning
In set setref, no reference to record dbkey-2 exists in the set occurrence of owner.
For backward chaining, member record dbkey-3 which should have contained the reference is also given.

FOR OWNER-DBKEY dbkey-1 SET-REF setref A RECORD-OCCURRENCE WAS DELETED FROM THE OWNER'S SET-CHAIN WITH MEMBER-DBKEY dbkey-2 (POINTING TO PRIOR DBKEY dbkey-3) WITH NO CORRESPONDING CHANGE OF A FORWARD-POINTER IN THE SET-CHAIN.

Meaning
In set setref, there exists in the set occurrence of owner dbkey-1 a reference to record dbkey-2 which is not, however, a member of the chain. For backward chaining, member record dbkey-3 which contains the reference is also given.

IN A HASH-OVERFLOW-CHAIN FOR KEYLENGTH key-length A POINTER TO NEXT OVERFLOW-PAGE actkey-1 WAS ADDED TO PAGE actkey-2 WITH NO CORRESPONDING CHANGE OF THE BACKWARD POINTER IN THE OVERFLOW-CHAIN.

Meaning
In a hash overflow chain of the CALC key with key length key-length, there exists in page actkey-2 a forward pointer to page actkey-1 with no corresponding backward pointer in the chain of overflow pages.
IN A HASH-OVERFLOW-CHAIN FOR KEYLENGTH \textit{key-length} A POINTER TO NEXT OVERFLOW-PAGE \textit{actkey-1} WAS DELETED FROM PAGE \textit{actkey-2} WITH NO CORRESPONDING CHANGE OF THE BACKWARD POINTER IN THE OVERFLOW-CHAIN.

\textbf{Meaning}
In a hash overflow chain of the CALC key with key length \textit{key-length}, there exists in page \textit{actkey-1} a backward pointer to page \textit{actkey-2} with no corresponding forward pointer in the chain of overflow pages.

IN A HASH-OVERFLOW-CHAIN FOR KEYLENGTH \textit{key-length} A POINTER TO PRIOR OVERFLOW-PAGE \textit{actkey-1} WAS ADDED TO PAGE \textit{actkey-2} WITH NO CORRESPONDING CHANGE OF THE FORWARD POINTER IN THE OVERFLOW-CHAIN.

\textbf{Meaning}
In a hash overflow chain of the CALC key with key length \textit{key-length}, there exists in page \textit{actkey-2} a backward pointer to page \textit{actkey-1} with no corresponding forward pointer in the chain of overflow pages.

IN A HASH-OVERFLOW-CHAIN FOR KEYLENGTH \textit{key-length} A POINTER TO PRIOR OVERFLOW-PAGE \textit{actkey-1} WAS DELETED FROM PAGE \textit{actkey-2} WITH NO CORRESPONDING CHANGE OF THE FORWARD POINTER IN THE OVERFLOW-CHAIN.

\textbf{Meaning}
In a hash overflow chain of the CALC key with key length \textit{key-length}, there exists in page \textit{actkey-1} a forward pointer to page \textit{actkey-2} with no corresponding backward pointer in the chain of overflow pages.

IN THE OVERFLOW-CHAIN OF A DUPLICATE TABLE WITH MAIN LEVEL \textit{actkey-1} DBTT-COL \textit{col-no} A BACKWARD POINTER TO PAGE \textit{actkey-2} WAS ADDED TO PAGE \textit{actkey-3} WITH NO CORRESPONDING CHANGE IN THE PREDECESSOR.

\textbf{Meaning}
In the overflow chain of a duplicate table with the main level in page \textit{actkey-1} and DBTT column number \textit{col-no}, there exists in page \textit{actkey-3} a backward pointer to page \textit{actkey-2} with no corresponding forward pointer in the chain of overflow pages.

IN THE OVERFLOW-CHAIN OF A DUPLICATE TABLE WITH MAIN LEVEL \textit{actkey-1} DBTT-COL \textit{col-no} A BACKWARD POINTER TO PAGE \textit{actkey-2} WAS DELETED FROM \textit{actkey-3} WITH NO CORRESPONDING CHANGE IN THE PREDECESSOR.

\textbf{Meaning}
In the overflow chain of a duplicate table with the main level in page \textit{actkey-1} and DBTT column number \textit{col-no}, there exists in page \textit{actkey-3} a forward pointer to page \textit{actkey-2} with no corresponding backward pointer in the chain of overflow pages.
IN THE OVERFLOW-CHAIN OF A DUPLICATE TABLE WITH MAIN LEVEL actkey-1 DBTT-COL col-no A
FORWARD pointer TO PAGE actkey-2 WAS ADDED TO PAGE actkey-3 WITH NO
CORRESPONDING CHANGE IN THE SUCCESSOR.

Meaning
In the overflow chain of a duplicate table with the main level in page actkey-1 and DBTT column number col-no, there exists in page actkey-3 a forward pointer to page actkey-2 with no corresponding backward pointer in the chain of overflow pages.

IN THE OVERFLOW-CHAIN OF A DUPLICATE TABLE WITH MAIN LEVEL actkey-1 DBTT-COL col-no A
FORWARD pointer TO PAGE actkey-2 WAS DELETED FROM PAGE actkey-3 WITH NO
CORRESPONDING CHANGE IN THE SUCCESSOR.

Meaning
In the overflow chain of a duplicate table with the main level in page actkey-1 and DBTT column number col-no, there exists in page actkey-2 a backward pointer to page actkey-3 with no corresponding forward pointer in the chain of overflow pages.

3.7.4.2 Global inconsistencies (index check)

Errors in the index value relationships between two chained table pages of an indexed table or between the pages of the overflow chain of a duplicate table are reported by BCHECK with the header:

THE FOLLOWING CHANGE OF DATABASE IS INCONSISTENT DUE TO WRONG KEY VALUE
RELATIONS IN MATCHING CHECK-RECORDS

If, however, BCHECK reports the following in the header:

THE FOLLOWING CHANGE OF DATABASE IS INCONSISTENT DUE TO DIFFERENT KEY LENGTHS
IN MATCHING CHECK-RECORDS

then the index values to be compared were of different lengths. In such a case, BCHECK does not perform the actual index check.
Inconsistency messages

The index check messages are listed below in alphabetical order.

FOR OWNER-DBKEY dbkey DBTT-COL col-no a TABLE-OCURRENCE at TABLE-LEVEL levno was added to PAGE actkey-1 pointing to NEXT-TABLE in PAGE actkey-2 with no corresponding change of a backward-pointer in the TABLE-CHAIN.

WITH KEY AND/OR RSQ:
keyvalue-1
rsq-1

***THE FOLLOWING DATABASE-CHANGES MAY BE CONNECTED WITH THE INCONSISTENCY:

FOR OWNER-DBKEY dbkey DBTT-COL col-no a TABLE-OCURRENCE at TABLE-LEVEL levno was added to PAGE actkey-2 pointing to PRIOR-TABLE in PAGE actkey-1.

WITH KEY AND/OR RSQ:
keyvalue-2
rsq-2

Meaning

In the table chain which is uniquely identified by owner record dbkey and DBTT column number col-no, the highest index value keyvalue-1 rsq-1 in table page actkey-1 is not smaller than the lowest index value keyvalue-2 rsq-2 in successor page actkey-2 of the table chain at the same level levno.

FOR OWNER-DBKEY dbkey DBTT-COL col-no an ENTRY REFERRING TO A TABLE-OCURRENCE at LEVEL levno was added to the table at NEXT-HIGHER LEVEL pointing to a PAGE actkey-1 with the reference coming from PAGE actkey-2 with no corresponding change of location of the TABLE-OCURRENCE pointed at.

WITH KEY AND/OR RSQ:
keyvalue-1
rsq-1

***THE FOLLOWING DATABASE-CHANGES MAY BE CONNECTED WITH THE INCONSISTENCY:

FOR OWNER-DBKEY dbkey DBTT-COL col-no a TABLE-OCURRENCE at TABLE-LEVEL levno was added to PAGE actkey-1 pointing to NEXT-HIGHER TABLE in PAGE actkey-2.

WITH KEY AND/OR RSQ:
keyvalue-2
rsq-2

Meaning

In the indexed table which is uniquely identified by owner record dbkey and DBTT column number col-no, in table page actkey-2 of level levno+1 the index value keyvalue-1 rsq-1 of an entry is smaller than the highest index value keyvalue-2 rsq-2 in the table page actkey-1 of the next lower level levno to which it refers.
IN AN INDEXED TABLE FOR OWNER-DBKEY dbkey DBTT-COL col-no AN ENTRY AT TABLE-LEVEL levno WAS ADDED TO PAGE actkey-1 POINTING TO ITS PREDECESSOR IN THE TABLE-CHAIN actkey-2 WITH NO CORRESPONDING CHANGE OF THE TABLE AT NEXT-HIGHER LEVEL.

WITH KEY AND/OR RSQ:
keyvalue-1
rsq-1

***THE FOLLOWING DATABASE-CHANGES MAY BE CONNECTED WITH THE INCONSISTENCY:

IN AN INDEXED TABLE FOR OWNER-DBKEY dbkey DBTT-COL col-no AN ENTRY AT TABLE-LEVEL levno+1 WAS ADDED TO PAGE actkey-3 POINTING TO A TABLE AT NEXT-LOWER LEVEL actkey-2 WITH KEY AND/OR RSQ:
keyvalue-2
rsq-2

**Meaning**
In the indexed table which is uniquely identified by owner record dbkey and DBTT column number col-no, in table page actkey-3 of level levno+1 the index value keyvalue-2 rsq-2 of an entry is not smaller than the lowest index value keyvalue-1 rsq-1 of successor page actkey-1 of that table page actkey-2 to which the entry refers.

IN THE OVERFLOW-CHAIN OF A DUPLICATE-TABLE WITH MAIN LEVEL actkey-1 DBTT-COL col-no A BACKWARD POINTER TO PAGE actkey-2 WAS ADDED TO PAGE actkey-3 WITH NO CORRESPONDING CHANGE IN THE PREDECESSOR.

WITH RSQ:
rsq-1

***THE FOLLOWING DATABASE-CHANGES MAY BE CONNECTED WITH THE INCONSISTENCY:

IN THE OVERFLOW-CHAIN OF A DUPLICATE-TABLE WITH MAIN LEVEL actkey-1 DBTT-COL col-no A FORWARD POINTER TO PAGE actkey-3 WAS ADDED TO PAGE actkey-2.

WITH RSQ:
rsq-2

**Meaning**
In the overflow chain of the duplicates table with the main level in page actkey-1 and DBTT column number col-no, the highest RSQ value rsq-2 of the duplicates table in page actkey-2 is not smaller than the lowest RSQ value rsq-1 of successor page actkey-3 of the duplicates table.
3.7.4.3 Local inconsistencies

FATAL ERROR : THE \{OLD/NEW\} FPA-VALUE = value IS WRONG

Meaning
1. The central FPA value for an act-key-0 page, a DBTT page or a CALC page is non-zero.
2. A list takes up an entire page; the free space is less than one entry, but the remainder is more than 8 bytes for a page length of 2048 bytes or more than 12 bytes for a page length of 4000 or 8096 bytes.
3. The central FPA value is greater than the value calculated on the page.

\{INDIRECT CALC-RECORD/CALC-SEARCH-KEY\} recref/keyref CAN NOT BE CHECKED IN AREA realm-name AS CALC-SEARCH-KEY(S) AND / OR INDIRECT CALC-KEYS WITH SAME KEYLENGTH ARE NOT IN YOUR CHOICE

Meaning
The indirect CALC record or CALC SEARCH key cannot be checked, as there are a number of CALC records or CALC SEARCH keys of the same length in this realm.

ACT-KEY actkey IN SYSTEM-RECORD IS INVALID, COL-NR=col-no

Meaning
The act-key in a DBTT column of the SYSTEM record and the act-key of the page are different, or the reference to a table is missing.

ACT-KEY actkey IN DBTT-ENTRY dbttentry COLUMN col-no IS WRONG

Meaning
In one column of the DBTT entry, the realm ref is zero (record deleted) but the page number is greater than 1.

- DBTT-ENTRY FOR NON EXISTING RECORD

Meaning
In a column with a number higher than zero, the realm ref of a deleted record is non-zero.

- REALM-REF NOT ALLOWED TO SCHEMA

Meaning
The realm ref of the DBTT entry is higher than the permitted maximum number.

- REALM-REF INVALID FOR RECORD OR TABLE

Meaning
The realm ref of the DBTT entry identifies a realm which has not been released for storage.

- BLOCK-NUMBER TOO BIG

Meaning
The page number in the DBTT entry is higher than the permitted maximum.
BNR OF PPP IN TABLE-ENTRY te-no IS WRONG

**Meaning**
The page number in the (indirect CALC) table entry is zero, while the realm ref is non-zero.

BNR OF MAIN LEVEL-POINTER IN DTOB actkey IS IMPOSSIBLE

**Meaning**
The page number of the main level of a duplicates table overflow page is higher than the permitted maximum.

CHECK OF KEY keyref WITH DUPLICATE TABLE IS IMPOSSIBLE. REALM realm-name WHICH MAY CONTAIN OWNER-RECORD-OCCURRENCES AND ATTACHED DUPLICATE TABLES IS NOT SPECIFIED

**Meaning**
The duplicates table of a key which was to be checked is defined as “attached to owner”, and at least one owner occurs in a realm which is not user-selectable.

CHECK OF INDICES IS IMPOSSIBLE WITHOUT TOTAL CHECK OF THE DATABASE: THE INDEX CLAUSE IS IGNORED

**Meaning**
Indices cannot be checked as part of incremental checking.

COMPRESSION FORMAT IS INVALID

**Meaning**
The SIA record length differs from the length of a compression record as calculated locally in the page.

COMPRESSION-ENTRY c-e DOES NOT MATCH WITH RECORD-LENGTH

**Meaning**
The displacements noted in the record’s compression entry do not match the record length.

DB-KEY OF THE RECORD IS INVALID

**Meaning**
The DB key of the SYSTEM record and that of the SIA (anchor) are at variance.

DUMP OF {OLD/NEW} BLOCK-CONTENTS:

**Meaning**
Checking of the page contents of the “old” shadow database or of the “new” shadow database or the original has stopped: the page contents have therefore been dumped to printer.

DUPLICATE-TABLE-KEYLIST FOR TABLE-ENTRY te-no HAS AN INVALID LENGTH

**Meaning**
The length of the DT entry is not a multiple of 3 (for a page length of 2048 bytes) or 6 (for a page length of 4000 or 8096 bytes).
**Inconsistency messages**

**DUPLICATE-TABLE-KEY-LIST FOR TABLE-ENTRY** te-no IS OUT OF TABLE RANGE

**Meaning**
The DT entry is outside the duplicates table storage range.

**FIRST MEMBER DB-KEY** dbkey IST INVALID (SET-REF = setref, SCD-DISPL = displacement)

**Meaning**
The DB key of the first member record in the owner’s SCD is invalid.

**FPA VALUE OF {OLD/NEW} BLOCK** actkey IS IMPOSSIBLE

**Meaning**
The FPA value is greater than the page length minus the length of the page control information.

**FPA-VALUE FOR {OLD/NEW} {DBTT-ANCHORPAGE/DBTT-BLOCK/CALC-BLOCK/FPA-BLOCK/TABLE-BLOCK/DUPLICATE-TABLE-OVERFLOW-BLOCK}** actkey CLAIMS NO FULL PAGE, VALUE = fpa-value

**Meaning**
The central FPA value for a DBTT anchor page, DBTT page, CALC page, FPA page, table page or for a duplicates table overflow page is non-zero.

**FPA-VALUE(S) FOR {DBTT/CALC}-BLOCK(S)** pagenumber - pagenumber INDICATE(S) EMPTY PAGE(S) OR IS (ARE) NOT RECOGNIZED AS,{DBTT/CALC}-BLOCK(S), REALM IS realm-name

**Meaning**
The central FPA value for the DBTT pages or CALC pages is
X'07EC' for a page length of 2048 bytes,
X'0F8C' for a page length of 4000 bytes or
X'1F8C' for a page length of 8096 bytes
or the page layout is not the correct DBTT or CALC page layout.

**FREE-SPACE.DISPL IN THE BLOCKHEADER DOES NOT OCCUR AS SMALLEST RECORD-DISPL IN A BLOCK-Index**

**Meaning**
The FPA displacement in the page control information does not occur as the smallest record displacement in at least one page index.

**INCONSISTENT CALC-HEADER:** calc-header
- NUMBER OF ACTUAL ENTRIES value IS TOO BIG

**Meaning**
The number of current table entries is greater than the number of reserved entries.

- KEYLENGTH AND/OR #TABLE-ENTRIES DO NOT CORRESPOND WITH TABLE-LENGTH

**Meaning**
The table lengths calculated from the page control information using 'FREE SPACE.DISPL' and from the entries in the CALC table header do not match.
- **BLOCK-NUMBER OF SOME POINTERS ARE TOO BIG**

  **Meaning**
  The backward or forward pointer value in the CALC table header is greater than the permitted maximum.

- PRIMARY BUCKET $bnr$ IN REALM $realm-name$ HAS A BACKWARD-POINTER

  **Meaning**
  The backward pointer value for a primary bucket in the CALC table header is non-zero.

- **#ACTUAL-ENTRIES value DIFFERS FROM #PAGEINDEX-ENTRIES**

  **Meaning**
  The number of current table entries is not the same as the number of page indices on the direct CALC page.

**INCONSISTENT TABLE-HEADER: table-header**

- **NUMBER OF TABLE-ENTRIES DOES NOT CORRESPOND WITH TABLE LENGTH**

  **Meaning**
  The table length is not equal to the table length calculated from the number of reserved entries and the entry length.

- **NUMBER ACTUAL ENTRIES value TOO BIG**

  **Meaning**
  The number of current table entries is greater than the number of reserved entries.

- **TABLE-DESCRIPTION-BYTE DIFFERS FROM SIA**

  **Meaning**
  The table types entered in the table page header and in the SIA are not the same.

- **INDEX-LEVEL FOR A NON-INDEXED TABLE**

  **Meaning**
  A level higher than 0 is entered in the page header of a table other than an index table.

- **BLOCK-NUMBERS OF SOME POINTERS ARE TOO BIG**

  **Meaning**
  At least one pointer value in the table page header is greater than the permitted maximum.

**INCONSISTENT DUPLICATE-TABLE-HEADER: d-table-header**

- **DUPLICATE-TABLE-HEADER IS OUT OF TABLE-RANGE**

  **Meaning**
  The DT header value is outside the table storage range.
Inconsistency messages

- NUMBER OF FREE-TABLE-SPACE IS WRONG
  Meaning
  The FREE-TABLE.SPACE value in the DT header of an overflow page is at variance with
  the calculated free space.

- POINTER TO PRIOR OVERFLOW PAGE IN LEVEL-0 TABLE NOT ALLOWED
  Meaning
  The backward pointer value in the DT header of a primary page is non-zero.

- POINTER TO PRIOR OVERFLOW PAGE IN DTOB MISSING
  Meaning
  The backward pointer value in a DT header of an overflow page is zero.

- POINTER TO NEXT OVERFLOW PAGE IN A PAGE WITH MORE THAN ONE DATABASE KEY LIST
  NOT ALLOWED
  Meaning
  The forward pointer in the DT header is non-zero even though there is still at least one more
  duplicates table on the page.

- FREE SPACE IN DTOB NOT ALLOWED BETWEEN BLOCK-INDEX AND DT-HEADER
  Meaning
  The distribution table header of an overflow page does not come immediately after the page
  index.

INCORRECT act-key-0/act-key-N OF REALM realm-name:
- REALMNAME realm-name DIFFERS FROM SIA
  Meaning
  The realm name in the act-key-0 page or act-key-N page does not match the realm name
  in the SIA.

- SYSTEM-BREAK HAD OCCURRED
  Meaning
  The system break bit in the act-key-0 page or act-key-N page of the DBDIR is set.

- FPA-VALUE FOR act-key-0/act-key-N CLAIMS NO FULL PAGE, VALUE = fpa-value
  Meaning
  The FPA value for the act-key-0 page or act-key-N page is non-zero.

INDEX-POINTER OF TABLE-ENTRY te-no DOES NOT POINT TO A BLOCK-INDEX
  Meaning
  The displacement to the page index in the direct CALC table entry is outside the storage
  range of the page indices.
KEYLENGTH AND/OR #TABLE-ENTRIES DO NOT CORRESPOND WITH TABLE-LENGTH

**Meaning**
The table length calculated from the table header and the key length is at variance with the length calculated locally on the page.

LAST MEMBER DB-KEY *dbkey* IS INVALID (SET-REF = *setref*, SCD-DISPL = *displacement*)

**Meaning**
The DB key of the last member record in the owner's SCD is invalid.

MAIN LEVEL POINTER IN DTOB *actkey* REFERS BLOCK *pagenumber* OF MAIN LEVEL WHICH IS NO LEVEL-0 DUPLICATE TABLE-BLOCK TABLE-BLOCK WITH ONE INDEX-KEY

**Meaning**
Two or more records are stored on the main level of a duplicates table.

NEXT MEMBER DB-KEY *dbkey* IS INVALID (SET-REF = *setref*, SCD-DISPL = *displacement*)

**Meaning**
The DB key of the next member record in a member's SCD is invalid.

NUMBER OF PRESENT ENTRIES = 0 IN CALC-OVERFLOW-BLOCK *actkey*

**Meaning**
The number of table entries on a CALC overflow page is zero.

NUMBER OF ALLOWED ENTRIES = 0. THE CALC-TABLE WILL NOT BE ANALYZED

**Meaning**
The maximum number of entries in the table is set to 0, so the table cannot be used.

OWNER SEQUENCE NUMBER *number* IS INVALID (SET-REF = *setref*, SCD-DISPL = *displacement*)

**Meaning**
The owner's RSQ in the SCD is higher than the permitted maximum.

PPP *pointer* IS INVALID, REALM IS WRONG (SET-REF = *setref*, SCD-DISPL = *displacement*)

**Meaning**
The specified pointer contains either an area reference not defined in the schema or it points to a realm in which the record is not permitted (severity EASY LOCAL ERROR).

PRIOR MEMBER DB-KEY *dbkey* IS INVALID (SET-REF = *setref*, SCD-DISPL = *displacement*)

**Meaning**
The DB key of the preceding member record in a member's SCD is invalid.

REALM-REF OF PPP TABLE-ENTRY *te-no* IS WRONG

**Meaning**
The realm ref in the (indirect CALC) table entry is zero or is higher than the permitted maximum.
Inconsistency messages

RECORD-LENGTH INVALID OR BLOCK-INDEX DISPL WRONG

**Meaning**
The record length in the SIA is different from the length calculated locally in the page, or the displacement in the page index is incorrect.

RSQ $rsq$ IN TABLE-ENTRY $te-no$ IS TOO BIG

**Meaning**
The RSQ in the (CALC) table entry is higher than the permitted maximum.

RSQ $rsq$ OF ENTRY $te-no$ FOR EQUAL USER-KEYS NOT IN SEQUENCE

**Meaning**
– The RSQs for identical CALC keys are not sorted in ascending sequence.
– A table other than a duplicates table contains more than one RSQ.

RSQ OF CALC-TABLE-ENTRY $te-no$ DIFFERS FROM BLOCK-INDEX

**Meaning**
In a direct CALC area reference is made to a page index with a differing RSQ.

RSQS OF DB-KEY-LIST IN DUPLICATE TABLE FOR TABLE-ENTRY $te-no$ NOT IN SEQUENCE

**Meaning**
The RSQs in the duplicates table are not sorted in ascending sequence.

SET RELATION IS INCONSISTENT (SET-REF = $setref$, SCD-DISPL = $displacement$)

**Meaning**
The SCD does not tally with the set description in the SIA.

TABLE WAS STORED INTO WRONG REALM

**Meaning**
The table is in a realm which is not as specified in the SIA.

THE {OLD/NEW} FPA-VALUE = $value$ IS WRONG

**Meaning**
A list takes up an entire page, but there is a remainder.

1. The central FPA indicates less free space than is actually available.
2. The remainder is more than one entry, and the central FPA value and the local FPA value are at variance.
THE BLOCK-HEADER IS INCONSISTENT explanation:
- WRONG ACT-KEY

  Meaning
  The page’s act-key is incorrect.

- Miscellaneous message extension

  Meaning
  Error in bytes 5-20, the name of the field is entered in explanation.

THE BLOCK-INDEX IS INCONSISTENT explanation
- DISPL IMPOSSIBLE

  Meaning
  - The displacement in the page index of the duplicates header of a duplicates table overflow page does not tally with the FPA value in the page control information.
  - The displacement in the page index is less than the local FPA value or is greater than the permitted maximum.

- DBTT-COL = 0 FOR TABLE-REC

  Meaning
  The DBTT column number in the page index of the duplicates table in a duplicates table overflow page is 0.

- WRONG REC-REF

  Meaning
  The rec ref (record reference) in the page index is higher than the permitted maximum.

- WRONG STATUS-BYTE

  Meaning
  1. The status byte in the page index is not 0,1,2,3 or 4.
  2. The status byte in page index of a SYSTEM record which is not a table indicates a non-zero value.

- STATUS-BYTE AND REC-REF DON'T MATCH

  Meaning
  The status byte in the page index points to a spanned record, but the rec ref is ≠ 1.

- NON-CALC-REC IN CALC-BUCKET

  Meaning
  The status byte in the page index of a page from the CALC bucket shows a value ≠ 3.

- REC-REFS FOR CALC-RECORDS NOT EQUAL

  Meaning
  The rec refs in the page indices of the CALC records are not all the same.
Inconsistency messages

- **CALC-REC IN NON-CALC-BUCKET**

  **Meaning**  
The status byte in the page index of a page outside the CALC bucket shows a value of 3.

- **RSQ IS 0**

  **Meaning**  
The RSQ in the page index indicates a value of 0.

- **DBTT-COL = 0 FOR TABLE-REC**

  **Meaning**  
The DBTT column number in the page index of a table record indicates a value of zero.

- **DISPL IMPOSSIBLE FOR TABLE-REC**

  **Meaning**  
The displacement in the page index of the table record is greater than the permitted maximum.

- **WRONG DBTT-COL OR STATUS**

  **Meaning**  
The status byte in the page index shows a value ≠ 3, but the DBTT column number indicates a value higher than 0.

- **WRONG RSQ FOR GIVEN REC-REF**

  **Meaning**  
The RSQ of a SYSTEM record is higher than the permitted maximum.

- **WRONG DBTT-COL FOR GIVEN TABLE-REC**

  **Meaning**  
The DBTT column number of a SYSTEM record is higher than the permitted maximum.

- **IMPOSSIBLE LENGTH**

  The specified record or table length is not possible.

  **THE DB-KEYS IN THE OWNER-RECORD, POINTING TO ITS SET DON'T MATCH (SET-REF = setref, SCD-DISPL = displacement)**

  **Meaning**  
  In the SCD of an owner of an empty set, only the DB key of the first member record or only the DB key of the last member record is the same as the DB key of the set.

  **THE LIST-TABLE DOES NOT CONTAIN ENOUGH ENTRIES**

  **Meaning**  
The number of entries in the table header of a list is lower than the number of current page indices.
THE TABLE APPEARS TWICE IN THE SAME BLOCK

**Meaning**
The DB key of a table record appears in at least two page indices.

USER-CALC-KEY OF ENTRY *te-no* WAS STORED INTO WRONG BUCKET (CORRECT BUCKET: *bucket-no*)

**Meaning**
A CALC record in a primary bucket or the first overflow bucket has been stored in the wrong bucket.

USER-KEY OF TABLE-ENTRY *te-no* IS NOT IN SEQUENCE

**Meaning**
1. The CALC keys are not sorted in ascending sequence.
2. The table entry is out of sequence in a sorted table.

USER-KEY OF TABLE-ENTRY *te-no* IS A NOT ALLOWED DUPLICATE

**Meaning**
1. The CALC key of this entry already exists in the same page, and duplicates are not allowed.
2. An entry in a table other than a duplicates table already exists once, and duplicates are not allowed.

**DBTT-ANCHORPAGE actkey INCONSISTENT: VALUE content FROM itemname IS WRONG**

**Meaning**
An error was discovered in a DBTT anchor page. The incorrect content of the underlying field is output together with the act-key of the DBTT anchor page.

**DBTT-ANCHORPAGES INCONSISTENT: VALUE content FROM itemname IS WRONG, RECORD IS recordname**

**Meaning**
An error was discovered during the processing of DBTT anchor pages. The incorrect content of the underlying field is output together with the record type name.

**DBTT-ANCHORPAGE actkey INCONSISTENT: VALUES content FROM itemnames DON'T MATCH**

**Meaning**
An error was discovered in a DBTT anchor page. The incompatible content of the two underlying fields is output together with the act-key of the DBTT anchor page.

**DBTT-ANCHORPAGES INCONSISTENT: VALUES content FROM itemnames DON'T MATCH, RECORD IS recordname**

**Meaning**
An error was discovered during the processing of DBTT anchor pages. The incompatible content of the two underlying fields is output together with the record type names.
Inconsistency messages

DBTT-ANCHORPAGES INCONSISTENT: THE DBTT-BLOCK(S) pagenumber1 – pagenumber2 OF RECORD recordname1 ARE NOT SEPARATED FROM THE DBTT-BLOCK(S) pagenumber3 – pagenumber4 OF RECORD recordname2

Meaning
An overlap between two DBTT areas was identified after the sorting of the DBTT areas administered in the DBTT anchor pages in accordance with the act-key of the first DBTT page in question.
3.7.5 Usage of job switches

When an error occurs BCHECK sets job switches (i.e. switches set using the BS2000 command or macro MODIFY-JOB-SWITCHES). You can use these job switches in your procedures, but you should only provide a control automatism in the case that no job switch is set. In any other case it is necessary to consult the SYSLST protocol.

The following table describes which switches BCHECK sets and which measures are necessary:

<table>
<thead>
<tr>
<th>Type of error</th>
<th>Job switch</th>
<th>Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>WARNING</td>
<td>30</td>
<td>see the “Messages” manual</td>
</tr>
<tr>
<td>ERROR</td>
<td>31</td>
<td>see the “Messages” manual</td>
</tr>
<tr>
<td>SYSTEM-ERROR</td>
<td>31</td>
<td>see the “Messages” manual</td>
</tr>
<tr>
<td>EASY LOCAL CONSISTENCY ERROR</td>
<td>26</td>
<td>Send error report to systems support; database operation is possible</td>
</tr>
<tr>
<td></td>
<td></td>
<td>without restrictions</td>
</tr>
<tr>
<td>FATAL LOCAL CONSISTENCY ERROR</td>
<td>27</td>
<td>Send error report to systems support; database operation may possible</td>
</tr>
<tr>
<td></td>
<td></td>
<td>depending on the kind of inconsistency. Please contact systems support.</td>
</tr>
<tr>
<td>GLOBAL CONSISTENCY ERROR</td>
<td>28</td>
<td>Send error report to systems support; database operation may possible</td>
</tr>
<tr>
<td></td>
<td></td>
<td>depending on the kind of inconsistency. Please contact systems support.</td>
</tr>
</tbody>
</table>

Table 11: Usage of job switches together in BCHECK
4 Printing the schema/subschema information area with BPSIA

BPSIA prints the schema SIA report or the subschema SSIA report in the form of a table. The SIA report may be of assistance to the user, for instance:

– when storing records in the database using BINILOAD,
– when printing out certain tables using BSTATUS, or
– when printing out the contents of the database using BPRECORD.

The SSIA report, by contrast, is intended more as an aid for the programmers of DB applications.

BPSIA can be used to print the following:

– the user schema and the user subschemata
– the compiler schema and the compiler subschema
– the PRIVACY-AND-IQF schema and the PRIVACY-AND-IQF subschema

In order to print a schema or subschema, BPSIA selects the appropriate Schema Information Area (SIA) or Subschema Information Area (SSIA) from the DBDIR and lists them in the form of tables.

A printout of the user SIA or a user SSIA in the same form can also be obtained by entering a DISPLAY statement at the time of SIA/SSIA generation with the BGSIA utility routine, or with the BGSSIA utility routine, respectively.

In addition to outputting the data to SYSLST, you can also output it to a file in CSV format. The use of CSV format facilitates the further processing of the data in other system environments (e.g. in spreadsheet applications). The output in CSV format is described in the manual “Database Operation“, section “Outputting database information in a neutral format“.
4.1 System environment

Figure 9: BPSIA system environment

BPSIA can also be started online. The DBDIR is then flagged as being inconsistent. A warning is issued, saying that the SYSTEM-BREAK-BIT in DBDIR is set.

When you run BPSIA online, you must assume that the data output is not current since the DBH has not copied all the data from the buffer to the database yet. In order to obtain as much current data as possible, you should force a database update just before the BPSIA run using the DAL command CHECKPOINT or NEW RLOG. However, the BPSIA output may still differ from the actual contents of the database if an update is running parallel to this task.
4.2 BPSIA statements

BPSIA recognizes the following statements:

<table>
<thead>
<tr>
<th>Statement</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>DISPLAY SCHEMA</td>
<td>Print a schema</td>
</tr>
<tr>
<td>DISPLAY SUBSCHEMA</td>
<td>Print a subschema</td>
</tr>
<tr>
<td>END</td>
<td>Terminate statement input</td>
</tr>
</tbody>
</table>

Table 12: BPSIA statements

The two DISPLAY statements are optional; they may be repeated as often as desired.

Print a schema (DISPLAY SCHEMA)

```
[DISPLAY [IN CSV [csv-filename]] SCHEMA schema-name]
```

IN CSV

BPSIA also outputs the data in CSV format.

`csv-filename`

Name of the file to which the data is to be output in CSV format. The specification of `csv-filename` is mandatory in the first IN CSV statement of a BPSIA run (e.g. DISPLAY IN CSV 'BPSIA.CSV' ...).

For a detailed description of CSV format output, see the manual “Database Operation”, section “Outputting database information in a neutral format”.

`schema-name`

Name of the schema whose SIA is to be printed; the options are:

- user-schema-name
- COMPILER-SCHEMA
- PRIVACY-AND-IQF-SCHEMA
Print a subschema (DISPLAY SUBSCHEMA)

```
[DISPLAY [IN CSV [csv-filename]] SUBSCHEMA subschema-name]
```

IN CSV

BPSIA also outputs the data in CSV format.

`csv-filename`

Name of the file to which the data is to be output in CSV format. The specification of `csv-filename` is mandatory in the first IN CSV statement of a BPSIA run (e.g. DISPLAY IN CSV 'BPSIA.CSV' ...).

For a detailed description of CSV format output, see the manual “Database Operation”, section “Outputting database information in a neutral format”.

`subschema-name`

Name of the subschema whose SSIA is to be printed; the options are:

- `user-subschema-name`
- `COMPILER-SUBSCHEMA`
- `PRIVACY-AND-IQF-SS`

Terminate statement input (END)

```
END
```
4.3 Command sequence for starting BPSIA

The command sequence described here is based on the assumption that UDS/SQL was installed with IMON (see the section “START commands of the UDS/SQL programs” in chapter 2 of the “Creation and Restructuring” manual).

01 /ADD-FILE-LINK LINK-NAME=DATABASE,
   FILE-NAME=[:catid:[$userid.]dbname.DBDIR[.copy-name]]
02 /SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL,VERSION=version,SCOPE=*TASK
03 /START-UDS-BPSIA
04 bpsia statements
05 END

01 In this case, specifying :catid: is permitted (see the “Database Operation” manual).
02 The version of the utility routine is selected. Specification of the version is generally recommended, since several UDS/SQL versions can be installed in parallel.
03 The UDS/SQL utility routine can also be started with the alias BPSIA.
04 A period is treated as an end criterion. It may be followed by another statement.
4.4 Description of the SIA report

The SIA report consists of a printout of an SIA in the form of a table. It contains the most important information from a schema.

SIA PRINT REPORT (general information)

*Example*

```plaintext
*** SIA PRINT REPORT ***
DATABASE ID             = XXXXXXXX
DATABASE NAME           = SHIPPING
DATABASE-NAVNAME-VERSION = 004.00
SCHEMA NAME             = MAIL-ORDERS
SCHEMA TYPE             = USER SCHEMA
SIA VALIDATION DATE     = 2004-08-12  15:40:31
LENGTH OF SIA           =    6576
MAXIMUM AREA REF        =     12
MAXIMUM RECORD REF      =     15
MAXIMUM SET REF         =     34
MAXIMUM KEY REF         =     19
TCUA LENGTH             =   3424
MAXIMUM RECORD LENGTH   =    580
MAXIMUM ENTRY LENGTH    =     52
MAXIMUM NR MEMBERSHIPS  =      4
MAXIMUM SPLIT PARAMETER =      5
LENGTH KEY-BIT-STRING   =      0
LENGTH PHYSICAL BLOCK   =   4000
FPA-ENTRIES MAIN-BLOCK  =   1990
```
Under the header SIA PRINT REPORT, BPSIA prints the following general information:

- **DATABASE ID**: Identification the database is stored under
- **DATABASE NAME**: Name of the database
- **SCHEMA NAME**: Name of the schema
- **SCHEMA TYPE**: Type of schema
- **SIA VALIDATION DATE**: Validation date of the schema (date and time)
- **LENGTH OF SIA**: Self-explanatory
- **MAXIMUM AREA REF**: Highest realm number in the user’s database; after reorganization, it is not necessarily identical with the actual number of realms
- **MAXIMUM RECORD REF**: Highest number of a record type in the user’s database; after reorganization, it is not necessarily identical with the actual number of record types
- **MAXIMUM SET REF**: Highest set number in the user’s database; after reorganization, it is not necessarily identical with the actual number of sets
- **MAXIMUM KEY REF**: Highest number of an ASC, DESC, or SEARCH key in the user’s database; after reorganization, it is not necessarily identical with the actual number of corresponding keys
- **TCUA LENGTH**: Length of the Transaction Currency Area of a subschema
- **MAXIMUM RECORD LENGTH**: Length of the longest record type in the schema including set connection data (SCD)
- **MAXIMUM ENTRY LENGTH**: Length of the longest key in the schema (CALC key, ASC or DESC key, or SEARCH key), incremented by 7 bytes and rounded up to an integral multiple of 4
MAXIMUM NR MEMBERSHIPS
The maximum number of sets in the schema containing the same record type as member record type

MAXIMUM SPLIT PARAMETER
Maximum value specified in the DYNAMIC REORGANIZATION clause of the SSL

LENGTH-KEY-BIT-STRING
Check byte for the MODIFY function; if 0 is specified, 4 bytes are reserved

LENGTH PHYSICAL BLOCK
Length of the database pages

FPA ENTRIES MAIN-BLOCK
Number of possible FPA entries per FPA base page
REFERENCE NUMBERS

Example

*** REFERENCE NUMBERS ***

11 AREAS : 1 3 4 5 6 7 8 9 10 11 12
15 RECORDS: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
34 SETS : 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34
19 KEYS : 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

Under the header REFERENCE NUMBERS, BPSIA provides a printout listing the total number of realms (areas), record types, records and keys defined in the schema and their respective numbers.

The numbers printed reflect the current status of the database. They need not necessarily be the maximum values printed under the header SIA PRINT REPORT (see page 122).

AREAS
1: Database directory
2: Database compiler realm
3,..: User realms

The database compiler realm is part of the compiler database; its reference number is therefore not included in the printout of a user SIA.

RECORDS
1: SSIA RECORD
2,..: User record types
### AREA INFORMATION

**Example**

<table>
<thead>
<tr>
<th>REF</th>
<th>AREA NAME</th>
<th>TEMP</th>
<th>D/T</th>
<th>FPA-BEGIN</th>
<th>ENTRIES</th>
<th>EXTENTS</th>
<th>INCR-ACT</th>
<th>CURRENT</th>
<th>FREE</th>
<th>SCAN</th>
<th>REUSE-FREE-SPACE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DATABASE-DIRECTORY</td>
<td>D</td>
<td>1</td>
<td>100</td>
<td>0</td>
<td>NO</td>
<td>49</td>
<td>50</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>CUSTOMER-ORDER-RLM</td>
<td>D</td>
<td>3</td>
<td>37</td>
<td>0</td>
<td>NO</td>
<td>13</td>
<td>23</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>PURCHASE-ORDER-RLM</td>
<td>D</td>
<td>4</td>
<td>60</td>
<td>0</td>
<td>NO</td>
<td>24</td>
<td>33</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>CLOTHING</td>
<td>D</td>
<td>5</td>
<td>54</td>
<td>0</td>
<td>NO</td>
<td>20</td>
<td>32</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>HOUSEHOLD-GOODS</td>
<td>D</td>
<td>6</td>
<td>24</td>
<td>0</td>
<td>NO</td>
<td>7</td>
<td>16</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>SPORTS-ARTICLES</td>
<td>D</td>
<td>7</td>
<td>45</td>
<td>0</td>
<td>NO</td>
<td>8</td>
<td>36</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>FOOD</td>
<td>D</td>
<td>8</td>
<td>18</td>
<td>0</td>
<td>NO</td>
<td>8</td>
<td>8</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>LEISURE</td>
<td>D</td>
<td>9</td>
<td>45</td>
<td>0</td>
<td>NO</td>
<td>8</td>
<td>36</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>STATIONERY</td>
<td>D</td>
<td>10</td>
<td>24</td>
<td>0</td>
<td>NO</td>
<td>6</td>
<td>17</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>ARTICLE-RLM</td>
<td>D</td>
<td>11</td>
<td>63</td>
<td>0</td>
<td>NO</td>
<td>46</td>
<td>15</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>SEARCH-RLM</td>
<td>*</td>
<td>D</td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Under the header AREA INFORMATION, BPSIA prints information on the realms of the database:

**REF**  Realm number

**AREA NAME**  Realm name

**TEMP**  Marker for temporary realms

*  Realm is temporary

**D/T**  Marker indicating whether the realm is stored on disk or on tape

D:  Disk
T:  Tape

**FPA BEGIN**  Address (realm number and page number) of the first page of the free place administration in this realm

**ENTRIES**  Total number of database pages in this realm

**EXTENTS**  Number of FPA extents in this realm

---

1 The fields in bold print contain the value '0' before the BFORMAT utility routine is run.
INCR-ACT
Denotes the online extensibility of this realm
YES: The online extensibility is activated for the realm.
NO: The online extensibility is not activated for the realm.

CURRENT
Number of the page in which free storage space was last located
0: Realm has not yet been accessed

FREE Number of completely free pages

SCAN Indicates whether the pages of the free place administration have been searched
for free space
1: for the first time or
2: repeatedly

REUSE-FREE-SPACE
Flag indicating whether the BMODTT statement SET REUSE-FREE-SPACE has
been specified for this realm

- not specified
\* specified

When a user schema or compiler schema is output, the FPA-BEGIN and ENTRIES
values for the DBDIR (line 1) may not be up-to-date. This may be the case after
BREORG has been executed, for example.
### RECORD WITHIN LIST

**Example**

```plaintext
*** RECORD WITHIN LIST ***

<table>
<thead>
<tr>
<th>REF</th>
<th>AREA-NAME</th>
<th>NR-WITHIN</th>
<th>LIST OF RECORDS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DATABASE-DIRECTORY</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>CUSTOMER-ORDER-RLM</td>
<td>4</td>
<td>2 3 4 5</td>
</tr>
<tr>
<td>4</td>
<td>PURCHASE-ORDER-RLM</td>
<td>3</td>
<td>13 14 15</td>
</tr>
<tr>
<td>5</td>
<td>CLOTHING</td>
<td>4</td>
<td>6 7 8 9</td>
</tr>
<tr>
<td>6</td>
<td>HOUSEHOLD-GOODS</td>
<td>5</td>
<td>6 7 8 9 10</td>
</tr>
<tr>
<td>7</td>
<td>SPORTS-ARTICLES</td>
<td>5</td>
<td>6 7 8 9 10</td>
</tr>
<tr>
<td>8</td>
<td>FOOD</td>
<td>4</td>
<td>6 7 8 9</td>
</tr>
<tr>
<td>9</td>
<td>LEISURE</td>
<td>4</td>
<td>6 7 8 9</td>
</tr>
<tr>
<td>10</td>
<td>STATIONERY</td>
<td>4</td>
<td>6 7 8 9</td>
</tr>
<tr>
<td>11</td>
<td>ARTICLE-RLM</td>
<td>2</td>
<td>11 12</td>
</tr>
<tr>
<td>12</td>
<td>SEARCH-RLM</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
```

Under the header RECORD WITHIN LIST, BPSIA prints out the record types contained in the individual realms.

**REF**  Realm number

**AREA-NAME**  Name of the realm

**NR-WITHIN**  Number of record types stored in the realm

**LIST OF RECORDS**  Numbers of the record types
**RECORD INFORMATION**

*Example*

<table>
<thead>
<tr>
<th>REF</th>
<th>RECORD-NAME</th>
<th>LOC-MODE</th>
<th>LENGTH</th>
<th>SYSINFO</th>
<th>COMPR</th>
<th>IMPL-SET</th>
<th>LIST-SET</th>
<th>LOC-VIA</th>
<th>OPT-CLAIM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SSIA-RECORD</td>
<td></td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>CUSTOMER</td>
<td>DIRECT</td>
<td>116</td>
<td>48</td>
<td></td>
<td></td>
<td></td>
<td>588</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>CST-ORDERS</td>
<td></td>
<td>17</td>
<td>6</td>
<td></td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>ORD-ITEM</td>
<td></td>
<td>54</td>
<td>46</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>INSTALLMENT</td>
<td></td>
<td>44</td>
<td>18</td>
<td>28</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>ART-TYPE</td>
<td></td>
<td>29</td>
<td>4</td>
<td>29</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>ART-SELECTION</td>
<td></td>
<td>25</td>
<td>0</td>
<td>30</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>ART-DESCR</td>
<td>CALC</td>
<td>580</td>
<td>20 V</td>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>ARTICLE</td>
<td>CALC</td>
<td>215</td>
<td>128</td>
<td>31</td>
<td>7</td>
<td>7</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>SUBSET</td>
<td></td>
<td>58</td>
<td>56</td>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>COLORS</td>
<td></td>
<td>22</td>
<td>0</td>
<td>32</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>MATERIALS</td>
<td></td>
<td>21</td>
<td>0</td>
<td>33</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>SUPPLIER</td>
<td>CALC</td>
<td>167</td>
<td>37</td>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>PURCHASE-ORDER</td>
<td></td>
<td>50</td>
<td>40</td>
<td></td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>P-ORD-ITEM</td>
<td></td>
<td>36</td>
<td>28</td>
<td></td>
<td>17</td>
<td>17</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

Under the header RECORD INFORMATION, BPSIA prints a table with the most important information on the individual record types.

**REF**  Number of the record type

**RECORD-NAME**  Name of the record type

**LOC-MODE**  Type of LOCATION MODE clause defined in DDL

- **CALC:**  LOCATION MODE IS CALC
- **DIRECT:**  LOCATION MODE IS DIRECT or DIRECT-LONG

**LENGTH**  Total length of the record type including set connection data (SCD)

**SYSINFO**  Length of set connection data (in bytes)

**COMPR**  Marker indicating whether the records of the record type are compressed

- **:*:**  Compression
- **V:**  The record type includes a variable-length item

**IMPL-SET**  Number of the implicit set of this record type if a SEARCH key has been defined at record type level
LIST-SET
Number of a set defined by MODE IS LIST which contains the record types as member record type

LOC-VIA
Number of a set for which MODE IS LIST or PLACEMENT OPTIMIZATION has been defined and which contains the record type as member record type

OPT-CLAIM
Number of bytes reserved when a record of this record type is stored (important only if the record type in question is included as an owner record within a set defined with PLACEMENT OPTIMIZATION or ATTACHED TO OWNER). The value specifies the number of bytes required to store the owner, the expected member and the administration information and tables.
DBTT INFORMATION

Under the header DBTT INFORMATION, BPSIA prints out two tables:

- The first table contains information on the database key translation tables (DBTTs) of the individual record types in the schema.
- The second table lists the sets in which the record types in the schema are member record types or owner record types.

*Example*

```
*** DBTT INFORMATION ***

<table>
<thead>
<tr>
<th>REF RECORD-NAME</th>
<th>ANCHOR</th>
<th>CURRENT</th>
<th>NR</th>
<th>EXTENTS</th>
<th>LENGTH</th>
<th>PER-BLOCK</th>
<th>NO REUSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 SSIA-RECORD</td>
<td>1</td>
<td>21</td>
<td>995</td>
<td>0</td>
<td>4</td>
<td>995</td>
<td></td>
</tr>
<tr>
<td>2 CUSTOMER</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>331</td>
<td>0</td>
<td>12</td>
<td>331</td>
</tr>
<tr>
<td>3 CST-ORDERS</td>
<td>3</td>
<td>4</td>
<td>1</td>
<td>1990</td>
<td>0</td>
<td>4</td>
<td>995</td>
</tr>
<tr>
<td>4 ORD-ITEM</td>
<td>3</td>
<td>6</td>
<td>1</td>
<td>1990</td>
<td>0</td>
<td>4</td>
<td>995</td>
</tr>
<tr>
<td>5 INSTALMENT</td>
<td>3</td>
<td>9</td>
<td>1</td>
<td>995</td>
<td>0</td>
<td>4</td>
<td>995</td>
</tr>
<tr>
<td>6 ART-TYPE</td>
<td>11</td>
<td>4</td>
<td>1</td>
<td>497</td>
<td>0</td>
<td>8</td>
<td>497</td>
</tr>
<tr>
<td>7 ART-SELECTION</td>
<td>11</td>
<td>4</td>
<td>5</td>
<td>497</td>
<td>0</td>
<td>8</td>
<td>497</td>
</tr>
<tr>
<td>8 ART-DESCR</td>
<td>11</td>
<td>6</td>
<td>13</td>
<td>497</td>
<td>0</td>
<td>8</td>
<td>497</td>
</tr>
<tr>
<td>9 ARTICLE</td>
<td>11</td>
<td>8</td>
<td>63</td>
<td>995</td>
<td>0</td>
<td>4</td>
<td>995</td>
</tr>
<tr>
<td>10 SUBSET</td>
<td>6</td>
<td>10</td>
<td>1</td>
<td>995</td>
<td>0</td>
<td>4</td>
<td>995</td>
</tr>
<tr>
<td>11 COLORS</td>
<td>11</td>
<td>12</td>
<td>10</td>
<td>995</td>
<td>0</td>
<td>4</td>
<td>995</td>
</tr>
<tr>
<td>12 MATERIALS</td>
<td>11</td>
<td>12</td>
<td>10</td>
<td>995</td>
<td>0</td>
<td>4</td>
<td>995</td>
</tr>
<tr>
<td>13 SUPPLIER</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>662</td>
<td>0</td>
<td>12</td>
<td>331</td>
</tr>
<tr>
<td>14 PURCHASE-ORDER</td>
<td>4</td>
<td>18</td>
<td>1</td>
<td>497</td>
<td>0</td>
<td>8</td>
<td>497</td>
</tr>
<tr>
<td>15 P-ORD-ITEM</td>
<td>4</td>
<td>20</td>
<td>1</td>
<td>995</td>
<td>0</td>
<td>4</td>
<td>995</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>REF RECORD-NAME</th>
<th>OWNERSHIPS</th>
<th>MEMBERSHIPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 CUSTOMER</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>3 CST-ORDERS</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>4 ORD-ITEM</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>5 INSTALLMENT</td>
<td>28</td>
<td>4</td>
</tr>
<tr>
<td>6 ART-TYPE</td>
<td>29</td>
<td></td>
</tr>
<tr>
<td>7 ART-SELECTION</td>
<td>6</td>
<td>30</td>
</tr>
<tr>
<td>8 ART-DESCR</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>9 ARTICLE</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>10 SUBSET</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>11 COLORS</td>
<td>28</td>
<td>4</td>
</tr>
<tr>
<td>12 MATERIALS</td>
<td>32</td>
<td>33</td>
</tr>
<tr>
<td>13 SUPPLIER</td>
<td>12</td>
<td>15</td>
</tr>
<tr>
<td>14 PURCHASE-ORDER</td>
<td>17</td>
<td>15</td>
</tr>
<tr>
<td>15 P-ORD-ITEM</td>
<td>14</td>
<td>17</td>
</tr>
</tbody>
</table>
```

1 The items printed in bold contain the value ‘0’ before the BFORMAT utility routine is run.
<table>
<thead>
<tr>
<th>REF</th>
<th>Number of a record type</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECORD-NAME</td>
<td>Name of the record type</td>
</tr>
<tr>
<td>ANCHOR</td>
<td>First page of the DBTT anchor table (realm number and page number)</td>
</tr>
<tr>
<td>CURRENT</td>
<td>Record sequence number of the last stored record of this record type</td>
</tr>
<tr>
<td>NR</td>
<td>Maximum number of records of this record type which can be stored taking into account the current size of the DBTT</td>
</tr>
<tr>
<td>EXTENTS</td>
<td>Number of DBTT extents that currently exist</td>
</tr>
<tr>
<td>LENGTH</td>
<td>Length of a DBTT entry</td>
</tr>
<tr>
<td>PER BLOCK</td>
<td>Number of DBTT entries that can be accommodated in one page</td>
</tr>
<tr>
<td>NO REUSE</td>
<td>Database keys of deleted records cannot be reused.</td>
</tr>
<tr>
<td></td>
<td>Database keys of deleted records can be reused.</td>
</tr>
<tr>
<td>OWNERSHIPS</td>
<td>Numbers of the sets in which the record type is owner</td>
</tr>
<tr>
<td>MEMBERSHIPS</td>
<td>Numbers of the sets in which the record type is member</td>
</tr>
</tbody>
</table>

When a user schema or compiler schema is output, the BEGIN, NR and LENGTH values may not be up to date. This may be the case after BREORG has been executed, for example.
**CALC INFORMATION**

*Example*

```
*** CALC INFORMATION ***

<table>
<thead>
<tr>
<th>REC</th>
<th>RECORD-NAME</th>
<th>PPP-BITS</th>
<th>PHYSICAL_CALC_INFO</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>ART-DESCR</td>
<td>*</td>
<td>40</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>ARTICLE</td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>SUPPLIER</td>
<td>*</td>
<td>35</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Under the header CALC INFORMATION, BPSIA prints out information on the record types defined with LOCATION MODE IS CALC.

**REC REF**
- Number of the record type

**RECORD-NAME**
- Name of the record type

**DIR**
- Marker indicating whether a direct or indirect hash area was created
  - *: Direct hash area

**DUPL**
- Marker indicating whether key values may or may not occur more than once
  - *: Duplicates allowed

**LENGTH**
- Total length of the CALC key

**HASH-REF**
- Marker for a user-defined hash routine; If no marker is printed in this column, the standard UDS/SQL hash routine is used.

---

1 The items printed in bold contain the value ‘0’ before the BFORMAT utility routine is run.
PPP-BITS
The significance of an * under the individual columns of this field is as follows:
O There is a PPP that was created on relocating the owner record.
M There is a PPP that was created on relocating the member record.
TAB BPGSIZE sets this bit if the table was defined with the option WITH PHYSICAL LINK.
It may be useful to perform a BREORG run.

PHYSICAL CALC INFO
Physical information on the hash area:

FIRST-BUCKET
Address (realm number and page number) of the first CALC page in the primary area

NR-BUCKETS
Number of CALC pages reserved for the primary area

Information listed under the header PHYSICAL CALC INFO is repeated by BPSIA for each realm in which a hash area for the relevant record type is located.
SET INFORMATION

Under the header SET INFORMATION, BPSIA prints out two tables containing information on the sets defined in the schema.

Example

<table>
<thead>
<tr>
<th>REF</th>
<th>SET-NAME</th>
<th>TYPE</th>
<th>MODE</th>
<th>ORDER</th>
<th>INSERT</th>
<th>REMOVE</th>
<th>SOS</th>
<th>INIT</th>
<th>INCR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CST-ORD-PLACED</td>
<td>PTRAY</td>
<td>SORT</td>
<td>AUTO</td>
<td>TRAN</td>
<td>OWN</td>
<td>10</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>CST-O-CONTENTS</td>
<td>PTRAY</td>
<td>SORT</td>
<td>AUTO</td>
<td>PERM</td>
<td>CUR</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>OUTSTANDING</td>
<td>CHAIN</td>
<td>LAST</td>
<td>AUTO</td>
<td>TRAN</td>
<td>OWN</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>HIRE-PURCHASE</td>
<td>CHAIN</td>
<td>LAST</td>
<td>AUTO</td>
<td>PERM</td>
<td>OWN</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>OFFER</td>
<td>PTRAY</td>
<td>SORT</td>
<td>AUTO</td>
<td>PERM</td>
<td>CUR</td>
<td>100</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>SHORT-LIST</td>
<td>PTRAY</td>
<td>SORT</td>
<td>AUTO</td>
<td>PERM</td>
<td>CUR</td>
<td>100</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>P-ORD-SPEC</td>
<td>LIST</td>
<td>SORT</td>
<td>AUTO</td>
<td>PERM</td>
<td>CUR</td>
<td>15</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>MIN-STOCK-LEVEL</td>
<td>SING</td>
<td>CH-PR</td>
<td>SORT</td>
<td>MANL</td>
<td>TRAN</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>CONTAINING</td>
<td>CHAIN</td>
<td>NEXT</td>
<td>AUTO</td>
<td>PERM</td>
<td>OWN</td>
<td>10</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>CONTAINED-IN</td>
<td>CH-PR</td>
<td>NEXT</td>
<td>AUTO</td>
<td>PERM</td>
<td>OWN</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>SUPPLIERS</td>
<td>SING</td>
<td>PTRAY</td>
<td>SORT</td>
<td>AUTO</td>
<td>PERM</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>ARTICLES-AVAILABLE</td>
<td>PTRAY</td>
<td>SORT</td>
<td>AUTO</td>
<td>PERM</td>
<td>CUR</td>
<td>500</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>ORDERED-ARTICLES</td>
<td>CHAIN</td>
<td>LAST</td>
<td>AUTO</td>
<td>PERM</td>
<td>OWN</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>REORDERED-ARTICLES</td>
<td>CHAIN</td>
<td>LAST</td>
<td>AUTO</td>
<td>PERM</td>
<td>OWN</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>P-ORD-PLACED</td>
<td>CHAIN</td>
<td>FIRST</td>
<td>MANL</td>
<td>PERM</td>
<td>CUR</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>P-ORD-RECEIVED</td>
<td>CHAIN</td>
<td>LAST</td>
<td>AUTO</td>
<td>PERM</td>
<td>CUR</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>P-ORD-CONTENTS</td>
<td>LIST</td>
<td>NEXT</td>
<td>AUTO</td>
<td>PERM</td>
<td>CUR</td>
<td>20</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>RESULT-SET</td>
<td>DYN</td>
<td>PTRAY</td>
<td>SORT</td>
<td>MANL</td>
<td>TRAN</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>LIMITED-SET</td>
<td>DYN</td>
<td>PTRAY</td>
<td>SORT</td>
<td>MANL</td>
<td>TRAN</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>IQL-DYN1</td>
<td>DYN</td>
<td>PTRAY</td>
<td>SORT</td>
<td>MANL</td>
<td>TRAN</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>IQL-DYN2</td>
<td>DYN</td>
<td>PTRAY</td>
<td>SORT</td>
<td>MANL</td>
<td>TRAN</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>IQL-DYN3</td>
<td>DYN</td>
<td>PTRAY</td>
<td>SORT</td>
<td>MANL</td>
<td>TRAN</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>IQL-DYN4</td>
<td>DYN</td>
<td>PTRAY</td>
<td>SORT</td>
<td>MANL</td>
<td>TRAN</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>IQL-DYN5</td>
<td>DYN</td>
<td>PTRAY</td>
<td>SORT</td>
<td>MANL</td>
<td>TRAN</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>IQL-DYN6</td>
<td>DYN</td>
<td>PTRAY</td>
<td>SORT</td>
<td>MANL</td>
<td>TRAN</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>IQL-DYN7</td>
<td>DYN</td>
<td>PTRAY</td>
<td>SORT</td>
<td>MANL</td>
<td>TRAN</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>IQL-DYN8</td>
<td>DYN</td>
<td>PTRAY</td>
<td>SORT</td>
<td>MANL</td>
<td>TRAN</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>SYS_INSTALLMENT</td>
<td>IMPL</td>
<td>SORT</td>
<td>AUTO</td>
<td>PERM</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>SYS_ART-TYPE</td>
<td>IMPL</td>
<td>SORT</td>
<td>AUTO</td>
<td>PERM</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>SYS_ART-SELECTION</td>
<td>IMPL</td>
<td>SORT</td>
<td>AUTO</td>
<td>PERM</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>SYS_ARTICLE</td>
<td>IMPL</td>
<td>SORT</td>
<td>AUTO</td>
<td>PERM</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>SYS_COLORS</td>
<td>IMPL</td>
<td>SORT</td>
<td>AUTO</td>
<td>PERM</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>SYS_MATERIALS</td>
<td>IMPL</td>
<td>SORT</td>
<td>AUTO</td>
<td>PERM</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>IMPLICIT_RESULT_SET</td>
<td>DYN</td>
<td>PTRAY</td>
<td>SORT</td>
<td>MANL</td>
<td>TRAN</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
The first table contains the following entries:

<table>
<thead>
<tr>
<th>REF</th>
<th>Set number</th>
</tr>
</thead>
<tbody>
<tr>
<td>SET NAME</td>
<td>Set name</td>
</tr>
<tr>
<td>TYPE</td>
<td>Set type</td>
</tr>
<tr>
<td>DYN:</td>
<td>Dynamic set</td>
</tr>
<tr>
<td>IMPL:</td>
<td>Implicit set</td>
</tr>
<tr>
<td>SING:</td>
<td>SYSTEM set (singular set)</td>
</tr>
<tr>
<td>Std:</td>
<td>Standard set</td>
</tr>
<tr>
<td>MODE</td>
<td>Set mode</td>
</tr>
<tr>
<td>CHAIN:</td>
<td>Chain</td>
</tr>
<tr>
<td>CH-PR:</td>
<td>Chain with backward chaining (CHAIN LINKED TO PRIOR)</td>
</tr>
<tr>
<td>LIST:</td>
<td>List</td>
</tr>
<tr>
<td>PTRAY:</td>
<td>Pointer array</td>
</tr>
<tr>
<td>IMPL:</td>
<td>Implicit set</td>
</tr>
<tr>
<td>ORDER</td>
<td>Sort sequence within the set occurrences of the set</td>
</tr>
<tr>
<td>INSERT</td>
<td>Insertion of new member records in the set</td>
</tr>
<tr>
<td>AUTO:</td>
<td>AUTOMATIC</td>
</tr>
<tr>
<td>MANL:</td>
<td>MANUAL</td>
</tr>
<tr>
<td>REMOVE</td>
<td>Type of set membership</td>
</tr>
<tr>
<td>PERM:</td>
<td>Permanent (MANDATORY member)</td>
</tr>
<tr>
<td>TRAN:</td>
<td>Transient (OPTIONAL member)</td>
</tr>
<tr>
<td>SOS</td>
<td>Set occurrence selection (in the case of non-singular sets only)</td>
</tr>
<tr>
<td>CUR:</td>
<td>THRU CURRENT OF SET</td>
</tr>
<tr>
<td>OWN:</td>
<td>THRU LOCATION MODE OF OWNER</td>
</tr>
<tr>
<td>INIT</td>
<td>Initial number of set occurrences according to the POPULATION clause for this set</td>
</tr>
<tr>
<td>INCR</td>
<td>Number of entries by which a set occurrence can be increased according to the INCREASE clause for this set</td>
</tr>
</tbody>
</table>
The second table contains the following entries:

<table>
<thead>
<tr>
<th>SET REF</th>
<th>Set number</th>
</tr>
</thead>
<tbody>
<tr>
<td>SET-NAME</td>
<td>Set name</td>
</tr>
<tr>
<td>OWNER</td>
<td>Number of the owner record type in the case of a non-singular set</td>
</tr>
<tr>
<td>0: indicates a singular set</td>
<td></td>
</tr>
<tr>
<td>MEMBER</td>
<td>Number of the member record type</td>
</tr>
<tr>
<td>0: indicates a dynamic set</td>
<td></td>
</tr>
</tbody>
</table>
PHYS LNK
Marker for additional link between member records and their owner record (physical linked to owner)
*: Additional link defined

OWNER RSQ
Within the set connection data of the member record type: displacement of the item with the record sequence number of the owner record relative to the beginning of the set connection data (see the “Design and Definition” manual).

SCD-DISPL
Displacement of the set connection data relative to the beginning of the record:
OWN  in the owner record type
MEM  in the member record type

SCD-LNGTH
Length of the set connection data of the set:
OWN  in the owner record type
MEM  in the member record type

CHAIN-LNK
Number of the next set with:
OWN  the same owner record type
MEM  the same member record type

ANCHOR-ACT
Address (realm number and page number) of the anchor record of a SYSTEM set. For dynamic sets, the realm number points to the temporary realm, since dynamic sets are stored in temporary realms.

ANCHOR-DBK
Database key of the anchor record of ANCHOR-ACT
(number of the record type = 0)
Under the header KEY INFORMATION (NO CALC-SEARCH KEYS), BPSIA prints out information about the keys defined on the record type level and set level in the schema. The table does not contain any information on CALC keys and CALC SEARCH keys (see page 133 and page 142).

**SET REF**

Number of the set to which the key belongs

---

1. The items printed in bold contain the value ‘0’ before the BFORMAT utility routine is run.
SET-NAME
Name of the set

KEY REF
Number of the key

TYPE
Type of key
ASC: ASCENDING key
DESC: DESCENDING key
SEARCH: SEARCH key (USING INDEX)
DBKEY: sorted by record sequence number

LENGTH
Overall length of the key item

DUPL NO
Marker indicating whether duplicate key values are allowed or not
*: Duplicates are not allowed

DUPL TABLE
Marker for duplicates tables
*: Duplicates tables set up (TYPE IS DATABASE-KEY-LIST)
*: No duplicates tables set up (TYPE IS REPEATED-KEY)

TABLE
Marker for table
*: Table set up

INDEX
Marker indicating whether a single-level or multi-level table has been set up
*: Multi-level table

ATT
Marker indicating whether a set occurrence is to be stored in close proximity to the owner
*: ATTACHED TO OWNER

LIST SET
Marker indicating whether the set occurrence table has been set up as a list
*: List

PPP-BITS
The significance of an * under the individual columns of this field is as follows:
O There is a PPP that was created on relocating the owner record.
M There is a PPP that was created on relocating the member record.
TAB BPGSIZE sets this bit if the table was defined with the option WITH PHYSICAL LINK.
It may be useful to perform a BREORG run.
SPLIT Number of pages specified in the REORGANIZATION clause
2: Default value; also applies when no table exists

DBTT COLUMN
Column number in the DBTT of the owner record type in which the address of the
table has been entered
0: No table

SSIA DISPL
Displacement of key description in the SSIA relative to the beginning of all key infor-
mation for this set

OWNER DISPL
Within the set connection data for this set in the owner record type:
displacement of the item containing the address of the table; only applicable if the
table has been defined with WITH PHYSICAL LINK option

TABLE-ACTKEY
For standard sets:
AREA: Number of the realm in which the table is stored (DETACHED WITHIN);
0: Table is stored in the realm of the owner record type (DETACHED or
ATTACHED)
BNR: No entry for standard sets

In the case of non-standard sets: address of the table
 AREA: Realm number
 BN R: Page number; 0 for dynamic sets
**CALC-SEARCH-KEY INFORMATION**

*Example*

```plaintext
*** CALC-SEARCH-KEY INFORMATION ***

<table>
<thead>
<tr>
<th>SET REF</th>
<th>SET-NAME</th>
<th>KEY REF</th>
<th>LNGTH</th>
<th>DUPL</th>
<th>HASH</th>
<th>PPP-BITS</th>
<th>SSIA</th>
<th>PHYSICAL_CALC_INFO</th>
</tr>
</thead>
<tbody>
<tr>
<td>29</td>
<td>SYS_ART-TYPE</td>
<td>12</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>11 14 1</td>
</tr>
<tr>
<td>31</td>
<td>SYS_ARTICLE</td>
<td>14</td>
<td>8</td>
<td>*</td>
<td></td>
<td></td>
<td>0</td>
<td>11 15 3</td>
</tr>
<tr>
<td>32</td>
<td>SYS_COLORS</td>
<td>16</td>
<td>20</td>
<td>*</td>
<td></td>
<td></td>
<td>0</td>
<td>11 29 1</td>
</tr>
<tr>
<td></td>
<td></td>
<td>17</td>
<td>2</td>
<td>*</td>
<td></td>
<td></td>
<td>28</td>
<td>11 30 1</td>
</tr>
</tbody>
</table>
```

Under the header CALC-SEARCH-KEY INFORMATION, BPSIA prints out information on the SEARCH keys which are defined on record or set level (for SYSTEM sets) by USING CALC:

**SET REF**
- Set number (implicit set or SYSTEM set)

**SET-NAME**
- Name of the set

**KEY REF**
- Number of the key

**LNGTH**
- Total length of the key

**DUPL**
- Marker indicating whether duplicate key values may occur or not
  - *: Duplicates not allowed

**HASH**
- Marker for the hash routine used
  - ..: Standard UDS/SQL hash routine
  - > 0: Number of user-defined hash routine

**PPP-BITS**
- The significance of an * under the individual columns of this field is as follows:
  - O: There is a PPP that was created on relocating the owner record.
  - M: There is a PPP that was created on relocating the member record.
  - TAB: BPGSIZE sets this bit if the table was defined with the option WITH PHYSICAL LINK.
SSIA DISPL
Displacement of this key description in the SSIA relative to the beginning of all key information for this set

PHYSICAL CALC INFO
Physical information on the hash area

FIRST-BUCKET
Address of the first CALC page reserved for table entries

NR-BUCKETS
Number of CALC pages reserved for table entries
4.5 Description of the SSIA report

The SSIA report is a copy of a subschema in the form of a table.

SSIA PRINT REPORT (general information)

Example

* BPSIA

*** SSIA PRINT REPORT ***

SUBSCHEMA NAME = ADMIN

SIA VALIDATION DATE = 2004-08-12 15:40:31

LENGTH OF SSIA = 4296

TCUA LENGTH = 2952

CRA LENGTH = 200

CRR LENGTH = 336

CRS LENGTH = 2240

MAXIMUM AREA REF = 12

MAXIMUM RECORD REF = 15

MAXIMUM SET REF = 34

NR AREAS = 10

NR RECORDS = 14

NR SETS = 34

Under the header SSIA PRINT REPORT, BPSIA prints out the following general information:

SUBSCHEMA NAME
Self-explanatory

SIA VALIDATION DATE
Validation date of the associated schema (with date and time)

LENGTH OF SSIA
Self-explanatory
TCUA LENGTH
Length of the transaction currency area of the subschema

CRA LENGTH
Length of CURRENT OF AREA table within the TCUA

CRR LENGTH
Length of CURRENT OF RECORD table within the TCUA

CRS LENGTH
Length of CURRENT OF SET table within the TCUA

MAXIMUM AREA REF
Highest realm number within the subschema

MAXIMUM RECORD REF
Highest record type number within the subschema

MAXIMUM SET REF
Highest set number within the subschema

NR AREAS
Number of user realms within the subschema

NR RECORDS
Number of record types within the subschema

NR SETS
Number of sets within the subschema
REFERENCE NUMBERS

Example

*** REFERENCE NUMBERS ***

AREAS : 3 4 5 6 7 8 9 10 11 12
RECORDS : 2 3 4 5 6 7 8 9 10 11 12 13 14 15
SETS : 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34
KEYS : 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19

Under the header REFERENCE NUMBERS, BPSIA lists the numbers of the realms, record types, sets and keys contained in the subschema.

The numbers in the printout reflect the current status of the database. They need not necessarily be the maximum values listed under the header SSIA PRINT REPORT.
### AREA INFORMATION

*Example*

*** AREA INFORMATION ***

<table>
<thead>
<tr>
<th>REF</th>
<th>AREA-NAME</th>
<th>CRA-DISPL</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>CUSTOMER-ORDER-RLM</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>PURCHASE-ORDER-RLM</td>
<td>20</td>
</tr>
<tr>
<td>5</td>
<td>CLOTHING</td>
<td>40</td>
</tr>
<tr>
<td>6</td>
<td>HOUSEHOLD-GOODS</td>
<td>60</td>
</tr>
<tr>
<td>7</td>
<td>SPORTS-ARTICLES</td>
<td>80</td>
</tr>
<tr>
<td>8</td>
<td>FOOD</td>
<td>100</td>
</tr>
<tr>
<td>9</td>
<td>LEISURE</td>
<td>120</td>
</tr>
<tr>
<td>10</td>
<td>STATIONERY</td>
<td>140</td>
</tr>
<tr>
<td>11</td>
<td>ARTICLE-RLM</td>
<td>160</td>
</tr>
<tr>
<td>12</td>
<td>SEARCH-RLM</td>
<td>180</td>
</tr>
</tbody>
</table>

Under the header AREA INFORMATION, BPSIA prints out information on the CURRENT-OF-AREA table of the TCUA.

The rest of the realm information is contained in the SIA report.

**REF**  Realm number

**AREA-NAME**  Name of the realm

**CRA-DISPL**  Displacement of the associated realm entry in the CURRENT OF AREA table of the TCUA relative to the beginning of the table
**RECORD INFORMATION**

*Example*

<table>
<thead>
<tr>
<th>DB-KEY-LOCATION AREA-WITHIN-LIST</th>
</tr>
</thead>
<tbody>
<tr>
<td>REC</td>
</tr>
<tr>
<td>CRR</td>
</tr>
<tr>
<td>NR</td>
</tr>
<tr>
<td>IMPL</td>
</tr>
<tr>
<td>IMPL</td>
</tr>
<tr>
<td>IMPL</td>
</tr>
<tr>
<td>AREA-ID</td>
</tr>
<tr>
<td>LIST-OF-AREAS</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>REF RECORD-NAME</th>
<th>REC DISPL</th>
<th>CRR DISPL</th>
</tr>
</thead>
<tbody>
<tr>
<td>REF RECORD-NAME</td>
<td>DISPL</td>
<td>DISPL</td>
</tr>
<tr>
<td>REF RECORD-NAME</td>
<td>KEYS</td>
<td>SET DISPL</td>
</tr>
<tr>
<td>REF RECORD-NAME</td>
<td>DISPL</td>
<td>DISPL</td>
</tr>
<tr>
<td>REF RECORD-NAME</td>
<td>DISPL</td>
<td>DISPL</td>
</tr>
</tbody>
</table>

2 CUSTOMER 0 0 0 0 60 3
3 CST-ORDERS 72 24 1 3
4 ORD-ITEM 88 48 1 3
5 INSTALLMENT 96 72 1 28 3
6 ART-TYPE 128 96 1 29 1056 5 6 7 8 9
7 ART-SELECTION 160 120 1 30 1086 5 6 7 8 9
8 ART-DESCR 192 144 3 1116 5 6 7 8 9
9 ARTICLE 752 168 7 31 1146 5 6 7 8 9
10 SUBSET 840 192 0 1176 6 7
11 COLORS 848 216 2 32 11
12 MATERIALS 872 240 2 33 11
13 SUPPLIER 896 264 2 4
14 PURCHASE-ORDER 1032 288 0 4
15 P-ORD-ITEM 1048 312 0 4

Under the header RECORD INFORMATION, BPSIA prints out information on the record types of the subschema.

**REC-REF**
Number of the record type

**RECORD-NAME**
Name of the record type

**REC DISPL**
Displacement of the record type within the UWA relative to the beginning of the RECORD AREA

**CRR DISPL**
Displacement of the record type entry in the CURRENT-OF-RECORD table of the TCUA relative to the beginning of the table

**NR KEYS**
Number of keys defined for the record type; the keys defined on set level are included by BPSIA only if they are contained in a subschema set in which the record type is a member record type
IMPL SET
   Number of the implicit set, if a SEARCH key has been defined on record type level

DB-KEY-LOCATION
   Is printed by BPSIA only if a record defined with LOCATION MODE IS DIRECT or DIRECT-LONG occurs in the schema:
   
   REC DISPL
   Displacement of the area containing the database key, relative to the beginning of the RECORD AREA; if the database key item is contained in the record type, this area is the record area itself

   ITEM DISPL
   Displacement of the database key item relative to REC DISPL

AREA-WITHIN-LIST
   Information on the realms in which records of the record type can be stored:
   
   AREA-ID DISPL
   Displacement of the AREA ID item in the WITHIN clause relative to the beginning of the RECORD AREA

   LIST-OF-AREAS
   Numbers of the realms in which – in accordance with the subschema description – records of the record type can be stored
CALC KEY INFORMATION

Example

*** CALC KEY INFORMATION ***

<table>
<thead>
<tr>
<th>REF</th>
<th>RECORD-NAME</th>
<th>NR-ITEMS</th>
<th>ITEM-REF</th>
<th>REC-DISPL</th>
<th>LENGTH</th>
<th>TYPE</th>
<th>NEXT-SET</th>
<th>NEXT-KEY</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>ARTICLE-DESCR</td>
<td>1</td>
<td>6</td>
<td>26</td>
<td>40</td>
<td>4</td>
<td>5</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>ARTICLE</td>
<td>3</td>
<td>0</td>
<td>128</td>
<td>6</td>
<td>5</td>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>134</td>
<td>2</td>
<td>5</td>
<td>7</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>54</td>
<td>182</td>
<td>2</td>
<td>5</td>
<td>7</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>SUPPLIER</td>
<td>2</td>
<td>0</td>
<td>37</td>
<td>5</td>
<td>5</td>
<td>11</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>42</td>
<td>30</td>
<td>4</td>
<td>11</td>
<td>8</td>
</tr>
</tbody>
</table>

Under the header CALC KEY INFORMATION, BPSIA prints out a table with information on the CALC keys (LOCATION MODE IS CALC) included in the subschema:

- **REF**: Number of the associated record type
- **RECORD-NAME**: Name of the record type
- **NR-ITEMS**: Number of items making up the CALC key
- **ITEM-REF**: Displacement of a CALC key item within the record type in accordance with the subschema format
- **REC-DISPL**: The item is not contained in the subschema format of the record type
- **REC-DISPL**: Displacement of a CALC key item within the database record (schema format including set connection data)
- **LENGTH**: Length of the CALC key item
- **TYPE**: Item type:
  - 0: Database key item
  - 1: Numeric item (packed)
  - 2: Binary item
  - 4: Alphanumeric item
  - 5: Numeric item (unpacked)
  - 8: Floating-point item
  - 15: Various item types (only when the CALC key is made up of several items)
NEXT-SET
Number of the next set in the subschema, in which this item is defined as a key item

NEXT-KEY
Number of the key from NEXT-SET

The entries from ITEM-REF to NEXT-KEY are repeated for each CALC key item (NR-ITEMS).

ITEM STRING LIST

Under the header ITEM STRING LIST, BPSIA prints out a table showing the differences between the subschema format and schema format of a record type.

An “Item String” is a series of items defined in the subschema format of the record type in the same contiguous order as those in the schema format. If the subschema format is exactly the same as the schema format, the subschema record is an item string.

Example

*** ITEM STRING LIST ***

<table>
<thead>
<tr>
<th>REF</th>
<th>RECORD-NAME</th>
<th>COMPL</th>
<th>USER-REC</th>
<th>DB-REC</th>
</tr>
</thead>
<tbody>
<tr>
<td>REC</td>
<td>RECID-NAME</td>
<td>REC</td>
<td>DISPL</td>
<td>DISPL</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>LENGTH</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>CUSTOMER</td>
<td>*</td>
<td>0</td>
<td>48</td>
</tr>
<tr>
<td>3</td>
<td>CST-ORDERS</td>
<td>*</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>ORD-ITEM</td>
<td>*</td>
<td>0</td>
<td>46</td>
</tr>
<tr>
<td>5</td>
<td>INSTALLMENT</td>
<td>*</td>
<td>0</td>
<td>18</td>
</tr>
<tr>
<td>6</td>
<td>ART-TYPE</td>
<td>*</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>ART-SELECTION</td>
<td>*</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>ART-DESCR</td>
<td>*</td>
<td>58</td>
<td>78</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>20</td>
</tr>
<tr>
<td>9</td>
<td>ARTICLE</td>
<td>*</td>
<td>0</td>
<td>128</td>
</tr>
<tr>
<td>10</td>
<td>SUBSET</td>
<td>*</td>
<td>0</td>
<td>56</td>
</tr>
<tr>
<td>11</td>
<td>COLORS</td>
<td>*</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>MATERIALS</td>
<td>*</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>SUPPLIER</td>
<td>*</td>
<td>0</td>
<td>37</td>
</tr>
<tr>
<td>14</td>
<td>PURCHASE-ORDER</td>
<td>*</td>
<td>0</td>
<td>40</td>
</tr>
<tr>
<td>15</td>
<td>P-ORD-ITEM</td>
<td>*</td>
<td>0</td>
<td>28</td>
</tr>
</tbody>
</table>
The table contains the following entries:

REC REF
   Number of the record type

RECORD-NAME
   Name of the record type

COMPL REC
   Marker for identical subschema and schema record:
   *:    Identical

USER-REC DISPL
   Displacement of an item string in the subschema record relative to the beginning of record (in descending order of displacement)

DB-REC DISPL
   Displacement of the item string relative to the beginning of the schema record including the set connection data

LENGTH
   Length of the item string in the subschema record
   V:    preceding the length indicates that the item in question is a variable item

The entries from USER-REC DISPL to LENGTH are repeated for each item string in the subschema record.
# KEY ITEM LIST

*Example*

```
*** KEY ITEM LIST ***

<table>
<thead>
<tr>
<th>REF RECORD-NAME</th>
<th>ITEM-REF</th>
<th>REC-DISPL</th>
<th>LENGTH</th>
<th>SET-REF</th>
<th>KEY-REF</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 CST-ORDERS</td>
<td>0</td>
<td>6</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>10</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>12</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>14</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>4 ORD-ITEM</td>
<td>0</td>
<td>46</td>
<td>2</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>5 INSTALLMENT</td>
<td>20</td>
<td>38</td>
<td>2</td>
<td>28</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>22</td>
<td>40</td>
<td>2</td>
<td>28</td>
<td>11</td>
</tr>
<tr>
<td></td>
<td>24</td>
<td>42</td>
<td>2</td>
<td>28</td>
<td>11</td>
</tr>
<tr>
<td>6 ARTICLE-TYPE</td>
<td>0</td>
<td>4</td>
<td>25</td>
<td>29</td>
<td>12</td>
</tr>
<tr>
<td>7 ARTICLE-SELECTION</td>
<td>0</td>
<td>0</td>
<td>25</td>
<td>30</td>
<td>13</td>
</tr>
<tr>
<td>8 ARTICLE-DESCR</td>
<td>6</td>
<td>26</td>
<td>40</td>
<td>12</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>136</td>
<td>40</td>
<td>12</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>31</td>
<td>15</td>
</tr>
<tr>
<td>9 ARTICLE</td>
<td>0</td>
<td>128</td>
<td>6</td>
<td>12</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>134</td>
<td>2</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>136</td>
<td>40</td>
<td>12</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>31</td>
<td>15</td>
</tr>
<tr>
<td></td>
<td>48</td>
<td>176</td>
<td>4</td>
<td>31</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>52</td>
<td>180</td>
<td>2</td>
<td>31</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>54</td>
<td>182</td>
<td>2</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>31</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>86</td>
<td>214</td>
<td>1</td>
<td>12</td>
<td>10</td>
</tr>
<tr>
<td>11 COLORS</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>32</td>
<td>17</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2</td>
<td>20</td>
<td>32</td>
<td>16</td>
</tr>
<tr>
<td>12 MATERIALS</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>33</td>
<td>18</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>1</td>
<td>20</td>
<td>33</td>
<td>19</td>
</tr>
<tr>
<td>13 SUPPLIER</td>
<td>0</td>
<td>37</td>
<td>5</td>
<td>11</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>42</td>
<td>30</td>
<td>11</td>
<td>8</td>
</tr>
</tbody>
</table>
```
Under the header KEY ITEM LIST, BPSIA prints out a table with information on all key items of the record types of the subschema.

**REF**  Number of the associated record type

**RECORD-NAME**  
Name of the associated record type

**ITEM-REF**  
Displacement of the key item relative to the beginning of the subschema record

**REC-DISPL**  
Displacement of the key item relative to the beginning of the schema record, including set connection data

**LENGTH**  
Length of the key item

**SET-REF**  
Numbers (in ascending order) of the sets in which the key item is also contained;

**CALC KEY ITEM**  
identifies the CALC key items of the LOCATION MODE clause, which are not related to a set or key and thus do not have any set numbers or key numbers

**KEY-REF**  
Number of the key to which the key item belongs
### SET INFORMATION

#### Example

---

*** SET INFORMATION ***

<table>
<thead>
<tr>
<th>SET</th>
<th>REF SET-NAME</th>
<th>NR</th>
<th>CRS</th>
<th>CRS-SORT</th>
<th>LOC</th>
<th>REC</th>
<th>ITEM</th>
<th>NR</th>
<th>ITEM DB-REC</th>
<th>ITEM</th>
<th>ITEM</th>
</tr>
</thead>
<tbody>
<tr>
<td>CST-ORD-PLACED</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>DIR</td>
<td>0</td>
<td>60</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CST-O-CONTENTS</td>
<td>2</td>
<td>1</td>
<td>80</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OUTSTANDING</td>
<td>3</td>
<td>0</td>
<td>160</td>
<td>0</td>
<td>DIR</td>
<td>0</td>
<td>60</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HIRE-PURCHASE</td>
<td>4</td>
<td>0</td>
<td>240</td>
<td>0</td>
<td>DIR</td>
<td>0</td>
<td>60</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OFFER</td>
<td>5</td>
<td>1</td>
<td>320</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SHORT-LIST</td>
<td>6</td>
<td>1</td>
<td>400</td>
<td>46</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P-ORD-SPEC</td>
<td>7</td>
<td>1</td>
<td>480</td>
<td>86</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MIN-STOCK-LEVEL</td>
<td>8</td>
<td>1</td>
<td>560</td>
<td>90</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CONTAINING</td>
<td>9</td>
<td>0</td>
<td>640</td>
<td>0</td>
<td>CALC</td>
<td>752</td>
<td>3</td>
<td>128</td>
<td></td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>CONTAINED-IN</td>
<td>10</td>
<td>0</td>
<td>720</td>
<td>0</td>
<td>CALC</td>
<td>* 1056</td>
<td>3</td>
<td>150</td>
<td>54</td>
<td>182</td>
<td>2</td>
</tr>
<tr>
<td>SUPPLIERS</td>
<td>11</td>
<td>1</td>
<td>800</td>
<td>100</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ARTICLES-AVAILABLE</td>
<td>12</td>
<td>2</td>
<td>880</td>
<td>135</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ORDERED-ARTICLES</td>
<td>13</td>
<td>0</td>
<td>960</td>
<td>0</td>
<td>CALC</td>
<td>752</td>
<td>3</td>
<td>128</td>
<td>54</td>
<td>182</td>
<td>2</td>
</tr>
<tr>
<td>REORDERED-ARTICLES</td>
<td>14</td>
<td>0</td>
<td>1040</td>
<td>0</td>
<td>CALC</td>
<td>752</td>
<td>3</td>
<td>128</td>
<td>54</td>
<td>182</td>
<td>2</td>
</tr>
<tr>
<td>P-ORD-PLACED</td>
<td>15</td>
<td>0</td>
<td>1120</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P-ORD-RECEIVED</td>
<td>16</td>
<td>0</td>
<td>1200</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P-ORD-CONTENTS</td>
<td>17</td>
<td>0</td>
<td>1280</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYS_INSTALLMENT</td>
<td>28</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYS_ARTICLE-TYPE</td>
<td>29</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYS_ARTICLE-SELECTION</td>
<td>30</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYS_ARTICLE</td>
<td>31</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYS_COLORS</td>
<td>32</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SYS_MATERIALS</td>
<td>33</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>IMPLICIT_RESULT_SET</td>
<td>34</td>
<td>0</td>
<td>2160</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---


Under the header SET INFORMATION, BPSIA prints out a table with information about all sets of the subschema.

**SET-REF**
Set number

**SET-NAME**
Name of the set

**NR KEYS**
Number of keys of the set's member record

**CRS DISPL**
Displacement of the set entry in the CURRENT OF SET table of the TCUA relative to the beginning of the table

**CRS-SORT KEY DISPL**
Displacement of the ASC key entry or DESC key entry relative to the beginning of the KEY AREA in the TCUA

**SOS-OWNER-INFO**
Information on the owner of the set, if the SET OCCURRENCE SELECTION clause has been specified with THRU LOCATION MODE OF OWNER in the DDL:

- **LOC OWN:**
- **LOCATION MODE** of the owner record type
- **CALC:** With hash routine
- **DIR:** DIRECT or DIRECT-LONG, i.e. with database key

**ALIAS**
*: ALIAS clause defined

**REC DISPL**
Displacement of the record type relative to the beginning of the RECORD AREA in which the item for locating the owner is situated

If *identifier* or the ALIAS clause has been specified, this is an implicit record containing all implicitly defined items.

**(DIR)** Only if the LOCATION MODE of the owner is DIRECT or DIRECT-LONG:

**ITEM DISPL**
Displacement of the database key item, relative to REC DISPL
(CALC)

Only if the LOCATION MODE of the owner is CALC:

NR ITEMS
Number of items making up the CALC key

ITEM REF
Displacement of the CALC key item within the subschema record.
If an ALIAS clause has been specified: displacement of the ALIAS item within the record for implicitly defined items

DB-REC-DISPL
Displacement of the CALC key item in the schema record including set connection data

ITEM LENGTH
Length of the CALC key item or of the ALIAS item including set connection data

ITEM TYPE
Type of the CALC key item or ALIAS item (see page 150)

SET READY LIST
Numbers of the realms which can be referenced when accessing via set
### KEY INFORMATION

**Example**

```markdown
*** KEY INFORMATION ***

<table>
<thead>
<tr>
<th>SET REF</th>
<th>SET-NAME</th>
<th>KEY NEXT</th>
<th>NR</th>
<th>ITEM DB-REC</th>
<th>ITEM</th>
<th>ITEM</th>
<th>KEY-CHAIN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>CST-ORD-PLACED</td>
<td>1</td>
<td>28</td>
<td>0</td>
<td>6</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>0</td>
<td>3</td>
<td>4</td>
<td>10</td>
<td>2 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td>12</td>
<td>2 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td>14</td>
<td>2 5</td>
</tr>
<tr>
<td>2</td>
<td>CST-O-CONTENTS</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>46</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>OFFER</td>
<td>4</td>
<td>0</td>
<td>1</td>
<td>26</td>
<td>40</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>SHORT-LIST</td>
<td>5</td>
<td>0</td>
<td>1</td>
<td>26</td>
<td>40</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>P-ORD-SPEC</td>
<td>6</td>
<td>0</td>
<td>2</td>
<td>134</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>54</td>
<td>182</td>
<td>2 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>8</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>MIN-STOCK-LEVEL</td>
<td>7</td>
<td>0</td>
<td>3</td>
<td>128</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td>134</td>
<td>2 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>54</td>
<td>182</td>
<td>2 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>31</td>
<td>14</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>SUPPLIERS</td>
<td>8</td>
<td>0</td>
<td>2</td>
<td>42</td>
<td>30</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>37</td>
<td>5</td>
</tr>
<tr>
<td>12</td>
<td>ARTICLES-AVAILABLE</td>
<td>9</td>
<td>28</td>
<td>1</td>
<td>136</td>
<td>40</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>0</td>
<td>1</td>
<td>86</td>
<td>214</td>
<td>1 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>11</td>
<td>0</td>
<td>3</td>
<td>38</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>22</td>
<td>40</td>
<td>2 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>24</td>
<td>42</td>
<td>2 5</td>
</tr>
<tr>
<td>28</td>
<td>SYS_INSTALLMENT</td>
<td>11</td>
<td>0</td>
<td>3</td>
<td>20</td>
<td>37</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>42</td>
<td>27</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>15</td>
<td>136</td>
<td>40 4</td>
</tr>
<tr>
<td>29</td>
<td>SYS_ARTICLE-TYPE</td>
<td>12</td>
<td>0</td>
<td>1</td>
<td>25</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>13</td>
<td>0</td>
<td>1</td>
<td>52</td>
<td>180</td>
<td>2 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>14</td>
<td>68</td>
<td>3</td>
<td>176</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>54</td>
<td>182</td>
<td>2 5</td>
</tr>
<tr>
<td>30</td>
<td>SYS_ARTICLE-SELECTION</td>
<td>15</td>
<td>0</td>
<td>1</td>
<td>136</td>
<td>40</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16</td>
<td>28</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>20 4</td>
</tr>
<tr>
<td>31</td>
<td>SYS_ARTICLE</td>
<td>17</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>18</td>
<td>28</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>19</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>20 4</td>
</tr>
<tr>
<td>32</td>
<td>SYS_COLORS</td>
<td>16</td>
<td>28</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>20 4</td>
</tr>
<tr>
<td>33</td>
<td>SYS_MATERIALS</td>
<td>17</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>2 5</td>
</tr>
<tr>
<td></td>
<td></td>
<td>18</td>
<td>28</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td>19</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>20 4</td>
</tr>
</tbody>
</table>
```

Under the header KEY INFORMATION, BPSIA prints out a table with information on all keys in the subschema except for those defined in the LOCATION MODE clause:

**SET REF**
- Number of the associated set

**SET-NAME**
- Name of the associated set
KEY REF
  Number of the key

NEXT KEY
  Displacement of the next key defined for this set within the SSIA
  0: The current key is the last key or the only key

DESC
  Marker indicating whether the key is ASCENDING or DESCENDING
  *: DESCENDING key

NR ITEMS
  Number of items making up the key

ITEM REF
  Displacement of each item within the subschema record
  65535: The item is not contained in the subschema format of the record

DB-REC DISPL
  Displacement of each item within the schema record including set connection data

ITEM LENGTH
  Self-explanatory

ITEM TYPE
  Type of item (see page 150)

KEY-CHAIN
  If the item is defined as a key item in other sets:
  SET: Is the number of the next set with the same key item
  KEY: Is the number of this key
5 Output relational schema information with BPSQLSIA

The data in a UDS/SQL database can be accessed in relational terms. BPSQLSIA prints the relational schema information for an existing UDS/SQL subschema that was defined in accordance with the CODASYL model. The relational schema information serves as a programming aid for the SQL user.

5.1 Overview

Relational access can, among others, be performed via the following SQL interfaces:

– program interfaces in COBOL programs (see the “SQL for UDS/SQL” manual)
– DRIVE V2.1 (see the “DRIVE/WINDOWS (BS2000)” manuals)

To aid the SQL user when working in this way, BPSQLSIA can be used to print out a relational representation of existing UDS/SQL data structures that have been defined in accordance with the CODASYL model. In the following sections, this relational description of the data structures will be called relational schema information. The CODASYL schema remains unchanged, however, and can continue to be used by CODASYL applications. The relational schema information includes all the information needed, such as table names and field definitions, to permit the SQL user to work with a CODASYL database on a relational basis. It also indicates whether an existing CODASYL subschema can be processed on a completely relational basis or whether such access is limited.

BPSQLSIA generates separate relational schema information for each CODASYL subschema.
5.2 System environment

BPSQLSIA can be run in parallel with database operations and is restartable.

If you are working with the DATABASE application area, you can call BPSQLSIA with the BS2000 command START-UDS-PRINT-SQLSIA or the alias BPSQLSIA.

The UDS/SQL user syntax file sets the SDF user prompting mode to EXPERT. You can change the user prompting with the command:

```
MODIFY-SDF-OPTIONS GUIDANCE=*EXPERT/*NO/*MAXIMUM/*MEDIUM/*MINIMUM
```
### 5.3 Prerequisites for SQL access to CODASYL definitions

In order for a CODASYL subschema to be processed on a completely relational basis with SQL it must satisfy the following conditions:

<table>
<thead>
<tr>
<th>Condition</th>
<th>SQL limitation if the condition is not satisfied</th>
</tr>
</thead>
<tbody>
<tr>
<td>The temporary realm must be present.</td>
<td>No SQL access permitted.</td>
</tr>
<tr>
<td>All sets whose member record types are included in the subschema must likewise be included in the subschema.</td>
<td>No SQL access permitted for the record type involved.</td>
</tr>
<tr>
<td>All ASCENDING/DESCENDING/CALC and SEARCH keys must be completely contained in the subschema.</td>
<td>No SQL access permitted for the record type involved.</td>
</tr>
<tr>
<td>Sets may not be defined with ORDER IS NEXT or ORDER IS PRIOR.</td>
<td>Neither INSERT nor UPDATE permitted for the member record type of the set.</td>
</tr>
<tr>
<td>Record types may not be distributed among multiple realms.</td>
<td>INSERT not permitted for the record type involved.</td>
</tr>
<tr>
<td>A record type may not be defined with LOCATION MODE IS DIRECT or DIRECT-LONG</td>
<td>INSERT not permitted for the record type involved.</td>
</tr>
<tr>
<td>A record type may not be defined with the SSL clause COMPRESSION.</td>
<td>UPDATE not permitted for the record type involved.</td>
</tr>
<tr>
<td>A record type may not contain variable length fields.</td>
<td>No SQL access permitted for the record type involved.</td>
</tr>
<tr>
<td>A record type may not include packed or unpacked numerical fields for which the following apply with respect to the number of storage locations and the scale factor: number of storage locations $&gt; 15$ or scale factor $&lt; 0$ or scale factor $&gt; \text{number of storage locations}$. A positive scale factor gives the number of positions to the right of the decimal point. A negative scale factor specifies how many zeroes UDS/SQL must add to the field contents when performing calculations.</td>
<td>No SQL access permitted for the field involved, and no INSERT permitted for the record type involved.</td>
</tr>
</tbody>
</table>

Table 13: Prerequisites for SQL access
5.4 BPSQLSIA statements

<table>
<thead>
<tr>
<th>Statement</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>END</td>
<td>Terminates input</td>
</tr>
<tr>
<td>OPEN-DATABASE</td>
<td>Opens the database</td>
</tr>
<tr>
<td>PRINT-RELATIONAL-SCHEMINFO</td>
<td>Selects subschemas</td>
</tr>
</tbody>
</table>

Table 14: BPSQLSIA statements

The individual statements of BPSQLSIA are described below in alphabetical order.

**Terminate input (END)**

This statement is used to terminate input and start the program run.

END

The END statement has no operands.
Open database (OPEN-DATABASE)

The OPEN-DATABASE statement must be given as the first statement if you have not
assigned the database with

/ADD-FILE-LINK LINK-NAME=DATABASE, −
     FILE-NAME=[:catid:][$userid.]dbname.DBDIR[.copy-name]

If you have already issued a ADD-FILE-LINK statement, the OPEN-DATABASE statement
is rejected as an error and not offered in the SDF mask.

The file link name remains in effect until it is released with the command REMOVE-FILE-
LINK. The OPEN-DATABASE statement, by contrast, is effective only till the end of the
BPSQLSIA run.

<table>
<thead>
<tr>
<th>OPEN-DATABASE</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATABASE-NAME = &lt;dbname&gt;</td>
</tr>
<tr>
<td>COPY-NAME = *NONE / &lt;copy-name&gt;</td>
</tr>
<tr>
<td>USER-IDENTIFICATION = *OWN / &lt;userid&gt;</td>
</tr>
</tbody>
</table>

DATABASE-NAME = <dbname>
Name of the database with which you wish to work.

COPY-NAME = *NONE
Selects the database original for processing.

COPY-NAME = <copy-name>
Selects the database copy with the specified copy name for processing.

USER-IDENTIFICATION = *OWN
BPSQLSIA runs under the same user ID as the one under which the database is cataloged.

USER-IDENTIFICATION = <userid>
User ID under which the database is cataloged. The user ID is specified without the `$`
character.

A database copy can be assigned explicitly with

/ADD-FILE-LINK LINK-NAME=DATABASE, −
     FILE-NAME=[:catid:][$userid.]dbname.DBDIR.copy-name

but may also be specified in the OPEN-DATABASE statement.
Select subschemas (PRINT-RELATIONAL-SCHEMAINFO)

Up to 30 subschemas can be explicitly specified in one BPSQLSIA run. With PRINT *ALL, however, BPSQLSIA generates relational schema information for all subschemas of the database, regardless of how many exist.

BPSQLSIA outputs the relational schema information in the order in which the subschemas appear in the COSSD, even if a different order is specified in the PRINT command.

The PRINT statement may be specified more than once.

<table>
<thead>
<tr>
<th>PRINT-RELATIONAL-SCHEMAINFO</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBSCHEMA-NAME = *ALL / *ALL-EXCEPT(...) / list-poss(20): &lt;subschema-name&gt;</td>
</tr>
</tbody>
</table>
   *ALL-EXCEPT(...)  |
      NAME = list-poss(20): <subschema-name> |

SUBSCHEMA-NAME = *ALL
Relational schema information is generated for all subschemas of the database. All further PRINT statements are ignored.

SUBSCHEMA-NAME = *ALL-EXCEPT(…)
Relational schema information is generated for all subschemas of the database except for those listed following *ALL-EXCEPT.

   NAME = list-poss(20): <subschema-name>
Names the subschemas for which no relational schema information is to be generated.

SUBSCHEMA-NAME = list-poss(20): <subschema-name>
Relational schema information is generated for all the named subschemas.
5.5 Command sequence to start BPSQLSIA

The command sequence described here is based on the assumption that UDS/SQL was installed with IMON (see the section “START commands of the UDS/SQL programs” in chapter 2 of the “Creation and Restructuring” manual).

01 /ADD–FILE–LINK LINK–NAME=DATABASE,
   FILE–NAME=[:catid:][$userid.]dbname.DBDIR[.copy–name]]
02 /SELECT–PRODUCT–VERSION PRODUCT–NAME=UDS-SQL,VERSION=version
03 /START–UDS–BPSQLSIA
04 //OPEN–DATABASE DATABASE–NAME = ...]
05 //PRINT–statements
06 //END

01, 04 You must use one of the two assignments for the database.
02 The version of the utility routine is selected.
   Specification of the version is generally recommended, since several UDS/SQL
   versions can be installed in parallel.
03 BPSQLSIA can be called from any user ID. The UDS/SQL utility routine can
   also be started with the alias BPSQLSIA or START-UDS-PRINT-SQLSIA.
5.6 Description of the output of BPSQLSIA

BPSQLSIA outputs to SYSLST:

1. Information on the base tables
   Heading: INFORMATION ABOUT RELATIONAL SCHEMA *subschema-name*
   This includes:
   - Descriptions of the fields of each base table, including field name, data type, null-value condition, default value and additional information. Additional information may be: PRIMARY KEY SYSTEMDEFINED, UNIQUE or REFERENCES...
   - A summary of all unique keys at the set level and all unique keys at the record type level consisting of more than one field (UNIQUE summary).
   - A summary of all simple and compound keys (INDEX summary).
     The INDEX output is divided in two for all keys (e.g. compound key): the left-hand column contains the indexes that SQL can use; the right-hand column contains the indexes as defined in UDS/SQL (e.g. decomposed into items).
     Keys that have not been fully taken over into the subschema are indicated. Missing key fields are identified by three question marks.

2. Table of all limitations
   Heading: SHORT INFORMATION ABOUT RELATIONAL SCHEMA
   This includes:
   - A message indicating whether or not the subschema can be processed with SQL.
   - If the subschema can be processed with SQL, a summary of the types of SQL access permitted for each base table, under the heading: SHORT INFORMATION ABOUT TABLES.

3. Messages, if one or more conditions that restrict SQL access are satisfied (see page 163)
   (Heading: DIAGNOSTIC SUMMARY FOR SUBSCHEMANAME *subschema-name*)
5.7 Conversion rules

The CODASYL definitions are converted to relational schema information according to the following rules.

<table>
<thead>
<tr>
<th>No.</th>
<th>CODASYL subschema</th>
<th>Relational schema description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Record type</td>
<td>Base table identified with TABLE .... Name of the base table: record-type-name</td>
</tr>
</tbody>
</table>
| 2   | Record type that is owner record type in at least one set | Primary key: record-type-name_ of data type INTEGER  
Additional information: PRIMARY KEY SYSTEMDEFINED  
Null-value condition: NOT NULL |
| 3   | Item of a record type                 | Field of the base table of the corresponding data type with null-value condition: NOT NULL  
Default value: 0 for numeric fields  
'_' for alphanumeric fields |
| 4   | Group item, repeating group           | Data type STRUCTURE                                                                           |
| 5   | Repeating factor                      | (repeating factor)                                                                             |
| 6   | Item of type DATABASE-KEY             | Field of the base table with the same name  
Data type: INTEGER  
Default value: 0 |
| 7   | Item of type DATABASE-KEY-LONG        | Field of the base table with the same name  
Data type: CHARACTER ; length 8  
Additional information: ATTRIBUTE item-name IS DEFINED AS DATABASE-KEY-LONG; default value: X'00..00' |
| 8   | One or more system sets not defined MANDATORY AUTOMATIC | A base table with the name SYSTEM and primary key SYSTEM_, of data type INTEGER |
| 9   | Set relationship                      | Foreign key in the base table that corresponds to the member record type  
Field name: set-name_  
Data type: INTEGER  
Reference condition: REFERENCES owner-record-type-name |
| 10  | Set: MANDATORY AUTOMATIC              | Foreign key with null-value condition: NOT NULL                                                |
| 11  | Set: MANDATORY MANUAL                 | Foreign key with null-value condition: NOT NULL ON UPDATE  
Default value: NULL |

Table 15: Conversion rules for BPSQLSIA (part 1 of 2)
### Conversion rules

The name of the CODASYL subschema becomes the name of the relational schema. Hyphens in names in the CODASYL schema are replaced by underscores in the derived names in the relational schema information. Condition names (level number 88) are not output in the relational schema information.

Examples of the individual rules can be found on page 172ff.

The meanings of the relational terms and concepts are explained with examples in the “SQL for UDS/SQL” manual.

<table>
<thead>
<tr>
<th>No.</th>
<th>CODASYL subschema</th>
<th>Relational schema description</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>Set: OPTIONAL AUTOMATIC</td>
<td>Foreign key with null-value condition: NOT NULL ON INSERT</td>
</tr>
<tr>
<td>13</td>
<td>Set: OPTIONAL MANUAL</td>
<td>Foreign key without null-value condition Default value: NULL</td>
</tr>
<tr>
<td>14</td>
<td>Key consisting of one or more items. e.g. for compound keys:</td>
<td>Additional information: left-hand column: INDEX ([set-name_]group-name) right-hand column: INDEX ([set-name_]field-1,...,field-n)</td>
</tr>
<tr>
<td>15</td>
<td>Unique key at record type level consisting of one item</td>
<td>Additional information UNIQUE for the key field</td>
</tr>
<tr>
<td>16</td>
<td>Unique key at set level consisting of one item for a SYSTEM set defined with MANDATORY AUTOMATIC</td>
<td>Additional information UNIQUE for the key field</td>
</tr>
<tr>
<td>17</td>
<td>Unique key at record type level consisting of multiple items</td>
<td>Additional information UNIQUE for the corresponding base table: UNIQUE (item-1,...,item-n)</td>
</tr>
<tr>
<td>18</td>
<td>Unique key at set level consisting of multiple items for a SYSTEM set defined with MANDATORY AUTOMATIC</td>
<td>Additional information UNIQUE for the base table corresponding to the member record type: UNIQUE (item-1,...,item-n)</td>
</tr>
<tr>
<td>19</td>
<td>Unique key at set level consisting of one or more items for a set defined with MANDATORY AUTOMATIC, or for a SYSTEM set not defined with MANDATORY AUTOMATIC</td>
<td>Additional information UNIQUE for the base table corresponding to the member record type: UNIQUE (set-name_,item-1,...,item-n) where set-name_ is the associated foreign key attribute</td>
</tr>
</tbody>
</table>

Table 15: Conversion rules for BPSQLSIA (part 2 of 2)
### 5.8 Summary of the SQL access permitted for each base table

For each CODASYL subschema processed, BPSQLSIA outputs a summary as follows:

*** SHORT INFORMATION ABOUT TABLES

<table>
<thead>
<tr>
<th>TABLE</th>
<th>RET</th>
<th>INS</th>
<th>UPD</th>
<th>ATR</th>
</tr>
</thead>
<tbody>
<tr>
<td>base-table-1</td>
<td>y/n</td>
<td>y/n</td>
<td>y/n</td>
<td>y/n</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
</tr>
<tr>
<td>base-table-n</td>
<td>y/n</td>
<td>y/n</td>
<td>y/n</td>
<td>y/n</td>
</tr>
</tbody>
</table>

**RET:**
- y: The SQL retrieval access SELECT is permitted for this base table.
- n: The SQL retrieval access SELECT is not permitted for this base table.

**INS:**
- y: The SQL access INSERT is permitted for this base table.
- n: The SQL access INSERT is not permitted for this base table.

**UPD:**
- y: The SQL access UPDATE is permitted for this base table.
- n: The SQL access UPDATE is not permitted for this base table.

**ATR:**
- y: SQL access is permitted for all fields (attributes) of this base table.
- n: In this base table there is at least one field (attribute) to which no SQL access is possible.
5.9 Example

BPSQLSIA execution

```
/ADD-FILE-LINK LINK-NAME=DATABASE,FILE-NAME=STAFF.DBDIR
/SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL, VERSION=02.4A00
/START-UDS-BPSQLSIA
***** START        BPSQLSIA     (UDS/SQL  V2.4  0400 )     2004-08-12   15:40:15
//PRINT-RELATIONAL-SCHMAINFO SUBSCHEMA-NAME=STAFF-DB
//END
***** DIAGNOSTIC SUMMARY FOR SUBSCHEMA STAFF-DB

NO ERRORS
+++++        4 WARNINGS

***** END OF DIAGNOSTIC SUMMARY
***** NR OF DATABASE ACCESSES  :           11
***** NORMAL END   BPSQLSIA     (UDS/SQL  V2.4  0400 )     2004-08-12   15:40:15
```

DDI of the CODASYL subschema

The numbers shown are the numbers of the conversion rules and refer to the corresponding parts in the relational schema information.

```
SCHEMA PERS-DB.
AREA PERS-DB-REALM.
AREA TEMPO TEMPORARY.

1/2) — RECORD NAME IS DEPARTMENT WITHIN PERS-DB-REALM.

3) ———— 02 NAME        PIC X(30).
     02 LOCATION     PIC X(30).
     02 EXTERN       PIC X.
     02 DEPARTMENT-MGR TYPE IS BIN 31.

RECORD NAME IS EMPLOYEE

15) ———— LOCATION MODE IS CALC USING PERSONNEL-NR

     DUPLICATES ARE NOT ALLOWED

     WITHIN PERS-DB-REALM

14) ———— SEARCH KEY IS M-NAME

     USING CALC

     DUPLICATES ARE ALLOWED

17) ———— SEARCH KEY IS P-CODE,CITY,STREET

     USING INDEX

     DUPLICATES ARE NOT ALLOWED.
```
03 E-NAME PIC X(30).
5) 03 FIRSTNAME PIC X(30) OCCURS 5.
03 E-AGE TYPE IS BIN 15.
03 MARITAL-STATUS PIC X.
4) 03 CHILDREN OCCURS 10.
04 C-NAME PIC X(30).
04 C-AGE TYPE IS BIN 15.
03 ADDRESS OCCURS 2.
04 P-CODE PIC X(5).
04 CITY PIC X(15).
04 STREET PIC X(30).
03 PERSONNEL-NO TYPE IS BIN 31.
03 OCCUPATION PIC X(10).
03 SALARY PIC S9(8)V9(2).
03 BONUSES PIC S9(8)V9(2).
03 MGR-NO TYPE IS BIN 31.

RECORD NAME IS PROJECT WITHIN PERS-DB-REALM.

02 PROJ-NAME PIC X(30).
02 BUDGET PIC S9(10)V9(2).
02 PROJ-MGR TYPE IS BIN 31.

9)  SET NAME IS DEPT-EMP
     ORDER IS FIRST
     OWNER IS DEPARTMENT
10) --- MEMBER IS EMPLOYEE
     MANDATORY AUTOMATIC
     SEARCH KEY IS MGR-NO
     USING INDEX DUPLICATES NOT ALLOWED
19) --- SEARCH KEY IS OCCUPATION,SALARY
     USING INDEX DUPLICATES NOT ALLOWED
     SELECTION CURRENT.

SET NAME IS PROJ-EMP
ORDER IS FIRST
OWNER IS PROJECT.
MEMBER IS EMPLOYEE
13) --- OPTIONAL MANUAL
     SELECTION CURRENT.

8)  SET NAME IS INT-FUND
     ORDER IS FIRST
     OWNER IS SYSTEM.
     MEMBER IS PROJECT
12) --- OPTIONAL AUTOMATIC
     SEARCH KEY IS PROJ-NAME
     USING CALC DUPLICATES NOT ALLOWED.

SET NAME IS PROJ-EMP-2
ORDER IS FIRST
OWNER IS PROJECT.
MEMBER IS EMPLOYEE

11) MANDATORY MANUAL
SELECTION CURRENT.

SET NAME IS EXT-FUND
ORDER IS FIRST
OWNER IS SYSTEM.
MEMBER IS PROJECT
OPTIONAL MANUAL.

SET NAME IS SYS-EMP
ORDER IS FIRST
OWNER IS SYSTEM.
MEMBER IS EMPLOYEE
MANDATORY AUTOMATIC

16) SEARCH KEY SALARY
USING CALC DUPLICATES NOT ALLOWED

18) SEARCH KEY E-AGE,MARITAL-STATUS
USING INDEX DUPLICATES NOT ALLOWED.

Output of BPSQLSIA to SYSLST

The numbers shown are the numbers of the conversion rules and refer to the corresponding parts in the DDL.

INFORMATION ABOUT RELATIONAL SCHEMA STAFF

1) TABLE DEPARTMENT

<table>
<thead>
<tr>
<th>ATTRIBUTE</th>
<th>TYPE</th>
<th>NOT NULL</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2) DEPARTMENT_

3) D_NAME
<table>
<thead>
<tr>
<th>TYPE</th>
<th>NOT NULL</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>CHAR</td>
<td>NOT NULL</td>
<td>' '</td>
</tr>
</tbody>
</table>

4) D_LOCATION
| CHAR            | NOT NULL | ' '     |

5) EXTERN
| CHAR            | NOT NULL | ' '     |

6) DEPARTMENT_MGR
| INTEGER         | NOT NULL | 0       |

___________________________________________________________________________________

TABLE EMPLOYEE

<table>
<thead>
<tr>
<th>ATTRIBUTE</th>
<th>TYPE</th>
<th>NOT NULL</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5) FIRSTNAME
| CHAR          | NOT NULL | ' '     |

6) E_AGE
| SMALLINT      | NOT NULL | 0       |

7) MARITAL_STATUS
| CHAR          | NOT NULL | ' '     |

4) CHILDREN
| CHAR          | NOT NULL | ' '     |

8) C_NAME
| CHAR          | NOT NULL | ' '     |

9) C_AGE
| SMALLINT      | NOT NULL | 0       |

10) ADDRESS
| CHAR          | NOT NULL | ' '     |

11) P_CODE
| CHAR          | NOT NULL | ' '     |

12) CITY
<p>| CHAR          | NOT NULL | ' '     |</p>
<table>
<thead>
<tr>
<th>STREET</th>
<th>CHARACTER (30)</th>
<th>NOT NULL</th>
<th>' '</th>
</tr>
</thead>
<tbody>
<tr>
<td>PERSONNEL_NO</td>
<td>INTEGER</td>
<td>NOT NULL</td>
<td>0</td>
</tr>
<tr>
<td>OCCUPATION</td>
<td>CHARACTER (10)</td>
<td>NOT NULL</td>
<td>' '</td>
</tr>
<tr>
<td>SALARY</td>
<td>NUMERIC (10, 2)</td>
<td>NOT NULL</td>
<td>0</td>
</tr>
<tr>
<td>BONUSES</td>
<td>NUMERIC (10, 2)</td>
<td>NOT NULL</td>
<td>0</td>
</tr>
<tr>
<td>MGR_NO</td>
<td>INTEGER</td>
<td>NOT NULL</td>
<td>0</td>
</tr>
<tr>
<td>DEPT_EMP_</td>
<td>INTEGER</td>
<td>NOT NULL</td>
<td>REFERENCES DEPARTMENT</td>
</tr>
<tr>
<td>PROJ_EMP_</td>
<td>INTEGER</td>
<td>NULL</td>
<td>REFERENCES PROJECT</td>
</tr>
<tr>
<td>PROJ_EMP_2_</td>
<td>INTEGER</td>
<td>NOT NULL ON UPDATE NULL</td>
<td>REFERENCES PROJECT</td>
</tr>
</tbody>
</table>

INDEX TO BE USED BY SQL INDEX DEFINITION IN UDS

INDEX (PERSONNEL_NO)
INDEX (E_NAME)
INDEX (ADDRESS)
INDEX (DEPT_EMP_, MGR_NO)
INDEX (DEPT_EMP_, OCCUPATION, SALARY)
INDEX (P_CODE, CITY, STREET)
INDEX (DEPT_EMP_, MGR_NO)
INDEX (E_AGE, MARITAL-STATUS)
INDEX (E_AGE, MARITAL-STATUS)
INDEX (E_AGE, MARITAL-STATUS)
INDEX (SALARY)
INDEX (SALARY)
INDEX (SALARY)
INDEX (E_AGE, MARITAL-STATUS)

TABLE PROJECT

<table>
<thead>
<tr>
<th>ATTRIBUTE</th>
<th>TYPE</th>
<th>NOT NULL</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>PROJECT_</td>
<td>INTEGER</td>
<td>NOT NULL</td>
<td>PRIMARY KEY SYSTEMDEFINED</td>
</tr>
<tr>
<td>PROJ_NAME</td>
<td>CHARACTER (30)</td>
<td>NOT NULL</td>
<td>' '</td>
</tr>
<tr>
<td>BUDGET</td>
<td>NUMERIC (12, 2)</td>
<td>NOT NULL</td>
<td>0</td>
</tr>
<tr>
<td>PROJ_MGR</td>
<td>INTEGER</td>
<td>NOT NULL</td>
<td>0</td>
</tr>
</tbody>
</table>
12) ──

<table>
<thead>
<tr>
<th>ATTRIBUTE</th>
<th>TYPE</th>
<th>NOT NULL</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>INT_FUND_</td>
<td>INTEGER</td>
<td>NOT NULL ON INSERT</td>
<td>REFERENCES SYSTEM</td>
</tr>
<tr>
<td>EXT_FUND_</td>
<td>INTEGER</td>
<td>NULL</td>
<td>REFERENCES SYSTEM</td>
</tr>
</tbody>
</table>

UNIQUE (INT_FUND_,
    PROJ_NAME)

INDEX TO BE USED BY SQL
    INDEX DEFINITION IN UDS

INDEX (INT_FUND_,
    PROJ_NAME)

WARNING 4018 *** INDEX CAN BE USED ONLY WITHIN 'IN PREDICATE' OR WITHIN 'COMPARISON PREDICATE' WITH 'EQUALS OPERATOR'

TABLE SYSTEM

<table>
<thead>
<tr>
<th>ATTRIBUTE</th>
<th>TYPE</th>
<th>NOT NULL</th>
<th>DEFAULT</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM_</td>
<td>INTEGER</td>
<td>NOT NULL</td>
<td>0</td>
</tr>
</tbody>
</table>

*** SHORT INFORMATION ABOUT RELATIONAL SCHEMA

RELATIONAL SCHEMA CAN BE PROCESSED WITH SQL

*** SHORT INFORMATION ABOUT TABLES

RET = Y : TABLE CAN BE PROCESSED WITH SQL
    = N : TABLE CANNOT BE PROCESSED WITH SQL
INS = N : NO INSERT ALLOWED ON TABLE
    = N : NO UPDATE ALLOWED ON TABLE
ATR = Y : ALL ATTRIBUTES CAN BE PROCESSED WITH SQL

TABLE

DEPARTMENT
    Y Y Y Y
EMPLOYEE
    Y Y Y Y
PROJECT
    Y Y Y Y

***** DIAGNOSTIC SUMMARY FOR SUBSCHEMA STAFF

NO ERRORS
***** 4 WARNINGS

***** END OF DIAGNOSTIC SUMMARY
6 Printing statistics on the occupied storage space with BSTATUS

BSTATUS prints tabular overviews showing the status of a user’s realms. These overviews contain the following information:

– the free storage space in the realms,
– DBTT sizes and the number of possible/existing DBTT entries,
– the occupancy level of the tables and the storage space they require,
– the utilization of reserved hash areas and the number of overflow pages, and
– the distribution of record types over realms,

The status reports thus permit optimum use of the storage space reserved for the user’s realms.

In addition to outputting the data to SYSLST, you can also output it to a file in CSV format. The use of CSV format facilitates the further processing of the data in other system environments (e.g. in spreadsheet applications). The output in CSV format is described in the manual “Database Operation”, section “Outputting database information in a neutral format”.

6.1 Functions

The amount of storage space required for the data in the database varies during processing and is dependent on the nature of the DB applications storing and erasing records.

BSTATUS can be used to obtain an overview of the occupied storage space, giving the DB administrator complete control over storage space allocation. He is thus able to:

- adapt the storage space occupied by the realms of his database to immediate needs so as not to take up more space than is required
- prepare for DB applications which insert new records by allocating additional space to those DB elements (realms, tables) which have become too small to accommodate new records.

There is online access to the original database, i.e., in parallel with database operation, or to a shadow database.

When you run BSTATUS online, you must assume that the data output is not current since the DBH has not copied all the data from the buffer to the database yet. In order to obtain as much current data as possible, you should force a database update just before the BSTATUS run using the DAL command CHECKPOINT or NEW RLOG. However, the BSTATUS output may still differ from the actual contents of the database if an update is running parallel to this task.

The tables printed by BSTATUS supply the following specific information:

- Realm statistics - used/unused storage space per realm:
  - size of the realm in pages
  - number of unused pages
  - number of pages partially filled
  - number of full pages
  - total number of unused bytes in the realm
BSTATUS

Functions

- Set statistics - storage space (per set) occupied by tables
  - number of set occurrences
  - number of stored member records in the largest and smallest set occurrences, and average number of member records in each set occurrence

  The following information is recorded for each table in the set:
  - column numbers in the owner DBTT containing the addresses of the tables
  - filling ratio (= occupancy level) on index level 0 (main level)
  - filling ratio on all index levels other than the main level
  - maximum and average number of index levels other than the main level
  - number of set occurrences in which reorganization by BREORG results in a reduction in the number of index levels.

- Owner statistics - storage space (per set) occupied by the tables for one owner
  - number of member records
  - for set occurrence table of the owner, the DBTT column number containing the address of the table
  - filling ratio (= occupancy level) on index level 0 (main level)
  - filling ratio on all index levels other than the main level
  - number of index levels other than the main level
  - flag indicating whether BREORG can reduce the number of index levels.

- Record type statistics - used/unused DBTT entries per record type:
  - number of used DBTT entries, i.e. number of records stored
  - highest record sequence number
  - highest record sequence number possible, i.e. maximum number of records of this record type which can be stored
  - DBTT filling ratio as a percentage
**CALC key statistics** - the storage space used by the primary pages and overflow pages per hash area:

- number of reserved primary pages
- number of records (for a direct hash area) or number of pointers (for an indirect hash area) which can still be added
- number of empty primary pages
- occupancy level of the primary pages
- number of overflow pages
- number of records or pointers in the overflow pages
- occupancy level of the overflow pages
- depth factor, i.e. the average number of accesses required to locate a record

**Record number statistics** - the records of a record type stored in a realm

- for one or more specified realms: Number of records per record type which have been stored in this realm
- for one or more specified record types: Number of records of specified record types stored per realm

**BSTATUS** can also be used to obtain a printout of the statistics on the storage space occupied by

- the database directory (DBDIR) and/or
- the database compiler realm (DBCOM).
6.2 System environment

Figure 11: BSTATUS system environment
System environment

**Work files**

In order to buffer and sort the set and owner statistics, BSTATUS requires two work files. These work files are set up automatically on public disk by BSTATUS under the user ID under which it was started. The default link names of these files are SCRTCH1 and SORTWK.

**SCRTCH1**

BSTATUS requires this file as temporary storage for the set statistics when the output of set and owner statistics is requested.

**SORTWK**

Requires the SORT used by BSTATUS for sorting internal evaluation records (see manual “SORT (BS2000/OSD)”.

If the work files are to be created explicitly, they must be assigned the following attributes:

**Work-file-1**

File link name SCRTCH1

Access method = SAM

The data population for buffering can be calculated using the following formula:

\[ 132 \times (\text{no. of sets} + \text{no. of keys}) \text{ Bytes} \]

- **no. of sets**
  - Number of sets in the subschema to be checked

- **no. of keys**
  - Number of keys in the subschema to be checked

The primary allocation for Work-file-1 should be based on the data population that is to be buffered. There should always be an appropriate secondary allocation in case the storage space proves to be insufficient.
Work-file-2

SORT needs Work-file-2 if there is not enough virtual memory for pre-sorting. The primary allocation should be based on the data population that is to be sorted while taking account of the safety factor recommended by SORT (see the discussion of work files in the manual “SORT (BS2000/OSD)”). There should always be an appropriate secondary allocation in case it is necessary to extend the storage space.

File link name SORTWK

Access method=PAM

The data population for sorting can be calculated using the formula:

\[ 16 \times \text{no. of sort records} \text{ Bytes} \]

\text{no. of sort records}

Number of records processed in the table statistics.

If the two work files are not created explicitly, BSTATUS generates them automatically with the following names and sizes:

UTI.SAMWORK.tsn.time-stamp.nnnn (33,33)

UTI.tsn.SORTWK (120,120)

\text{tsn}

task sequence number of the current task.

\text{time-stamp}

date and time (yyyyymmddhhmmss) on which the file was created.

\text{nnnn}

four-digit sequential number.

If execution terminates normally, any work files created by BSTATUS and their file link names are deleted. Any work files that you have set up explicitly are not deleted and their file link names are not released.
6.3 BSTATUS statements

<table>
<thead>
<tr>
<th>Statement</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBSCHEMA</td>
<td>Print realm statistics</td>
</tr>
<tr>
<td>DISPLAY REALM</td>
<td>Designate subschema</td>
</tr>
<tr>
<td>DISPLAY TABLE FOR SET</td>
<td>Print set statistics</td>
</tr>
<tr>
<td>DISPLAY TABLE FOR OWNER</td>
<td>Print owner statistics</td>
</tr>
<tr>
<td>DISPLAY RECORD</td>
<td>Print record type statistics</td>
</tr>
<tr>
<td>DISPLAY CALC</td>
<td>Print CALC key statistics</td>
</tr>
<tr>
<td>DISPLAY RECORDNUMBER</td>
<td>Print record number statistics</td>
</tr>
<tr>
<td>END</td>
<td>Terminate statement input</td>
</tr>
</tbody>
</table>

Table 16: BSTATUS statements

All DISPLAY statements are optional. They can be specified in any sequence as often as desired.

Every BSTATUS statement may be terminated with a period (.).
Designate the subschema (SUBSCHEMA)

```
SUBSCHEMA IS subschema-name
```

The SUBSCHEMA statement must be the first statement entered. All realms, record types and sets whose statistics are to be printed out using BSTATUS must be contained in the specified subschema.
Print realm statistics (DISPLAY REALM)

DISPLAY [IN CSV [csv-filename]] REALM STATISTICS FOR \{realm-name-1,...\} \{ALL\}

IN CSV
BSTATUS also outputs the data in CSV format.

\textit{csv-filename}
Name of the file to which the data is to be output in CSV format. The specification of \textit{csv-filename} is mandatory in the first IN CSV statement of a BSTATUS run (e.g. DISPLAY IN CSV 'BSTATUS.CSV' ...).

For a detailed description of CSV format output, see the manual “Database Operation”, section “Outputting database information in a neutral format”.

\textit{realm-name}
Name of a realm for which realm statistics are to be printed.
You must specify names as follows:
– for user realms: the realm name defined in the Schema DDL by the AREA clause
– for the DBDIR: DATABASE DIRECTORY
– for the DBCOM: DATABASE COMPILER REALM

\textit{ALL}
BSTATUS prints realm statistics on all the realms contained in the specified subschema.
### Example

DISPLAY REALM STATISTICS FOR ALL

SPACE USED PER REALM

<table>
<thead>
<tr>
<th>REALMS</th>
<th>NR OF EMPTY PAGES</th>
<th>MAX NR OF CONTIGUOUS EMPTY PAGES</th>
<th>NR PAGES WITH FILLING PERCENTAGES BETWEEN 0-19</th>
<th>20-39</th>
<th>40-59</th>
<th>60-79</th>
<th>80-99</th>
<th>NR FULL USER PAGES</th>
<th>TOTAL USER NR OF OCTADS PAGES</th>
<th>FREE OCTADS TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUSTOMER-ORDER-RLM</td>
<td>23</td>
<td>23</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>10</td>
<td>37</td>
<td>95 500</td>
</tr>
<tr>
<td>PURCHASE-ORDER-RLM</td>
<td>33</td>
<td>33</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>23</td>
<td>60</td>
<td>135K</td>
</tr>
<tr>
<td>CLOTHING</td>
<td>21</td>
<td>20</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>6</td>
<td>42</td>
<td>101K</td>
</tr>
<tr>
<td>HOUSEHOLD-GOODS</td>
<td>15</td>
<td>14</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>24</td>
<td>59 700</td>
</tr>
<tr>
<td>SPORTS-ARTICLES</td>
<td>37</td>
<td>37</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>45</td>
<td>147K</td>
</tr>
<tr>
<td>FOOD</td>
<td>7</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>3</td>
<td>18</td>
<td>29 630</td>
</tr>
<tr>
<td>LEISURE</td>
<td>37</td>
<td>37</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>45</td>
<td>147K</td>
</tr>
<tr>
<td>STATIONERY</td>
<td>17</td>
<td>17</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>24</td>
<td>67 660</td>
</tr>
<tr>
<td>ARTICLE-RLM</td>
<td>1984</td>
<td>1984</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>41</td>
<td>2064</td>
<td>7 911K</td>
</tr>
<tr>
<td>SEARCH-RLM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOTALS</td>
<td>2 174</td>
<td>9</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>13</td>
<td>103</td>
<td>2 359</td>
<td>8 694K</td>
<td></td>
</tr>
</tbody>
</table>

BSTATUS does not provide statistics on temporary realms.

**REALMS**

Names of the realms

**NR OF EMPTY PAGES**

Number of empty pages

**MAX NR OF CONTIGUOUS EMPTY PAGES**

Maximum number of contiguous empty pages

**NR PAGES WITH FILLING PERCENTAGES BETWEEN**

Number of partially filled pages, divided into groups with the specified occupancy level (excluding FPA pages, Act-key-0 pages, and Act-key-N pages)
NR FULL USER PAGES
Number of full pages (excluding FPA pages, Act-key-0 pages, and Act-key-N pages)

TOTAL NR OF PAGES
Size of the realm in pages (including FPA pages, Act-key-0 pages, and Act-key-N pages)

FREE OCTADS TOTAL
Total number of unused bytes per realm
K: If the number is greater than 100,000 bytes, BSTATUS rounds it off to a multiple of 1,000 = 1 kbyte
M: If the number is greater than 100,000 kbytes, BSTATUS rounds it off to a multiple of 1,000 kbyte = 1 Mbyte

TOTALS
If ALL is specified in the DISPLAY REALM statement, BSTATUS prints the sums of all columns in the TOTALS line.
In this case it calculates the total number of unused bytes before rounding off the numbers for the individual realms.
Print set statistics (DISPLAY TABLE FOR SET)

`DISPLAY [IN CSV [csv-filename]] TABLE STATISTICS FOR SET [set-name-1,...] {*ALL[ EXCEPT setname-1,...]}`

IN CSV

BSTATUS also outputs the data in CSV format.

`csv-filename`
Name of the file to which the data is to be output in CSV format. The specification of `csv-filename` is mandatory in the first IN CSV statement of a BSTATUS run (e.g. DISPLAY IN CSV 'BSTATUS.CSV' ...).

For a detailed description of CSV format output, see the manual “Database Operation”, section “Outputting database information in a neutral format”.

`set-name`
Name of the set for which statistics are to be printed

*ALL  Statistics on all sets of the subschema are to be printed

*ALL EXCEPT `set-name-1`,...  Statistics are to be printed for all sets other than those named at `set-name`. 
### Example

**DISPLAY TABLE STATISTICS FOR SET *ALL**

<table>
<thead>
<tr>
<th>SET IDENTIFICATION</th>
<th>OWNER OCCURRENCE</th>
<th>MEMBER OCCURRENCE</th>
<th>TABLE STATISTICS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CONTAINING</strong></td>
<td>SINGULAR</td>
<td></td>
<td>&quot;***** EMPTY SET *****&quot;</td>
</tr>
<tr>
<td><strong>CONTAINED-IN</strong></td>
<td>SINGULAR</td>
<td></td>
<td>&quot;***** EMPTY SET *****&quot;</td>
</tr>
<tr>
<td><strong>SUPPLIERS</strong></td>
<td>SINGULAR</td>
<td></td>
<td>&quot;***** EMPTY SET *****&quot;</td>
</tr>
<tr>
<td><strong>ARTICLES-AVAILABLE</strong></td>
<td>1</td>
<td>63</td>
<td>63</td>
</tr>
<tr>
<td><strong>P-ORD-SPEC</strong></td>
<td>13</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td><strong>SHORT-LIST</strong></td>
<td>5</td>
<td>1</td>
<td>6</td>
</tr>
<tr>
<td><strong>OFFER</strong></td>
<td>4</td>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td><strong>P-ORD-CONTENTS</strong></td>
<td></td>
<td></td>
<td>&quot;***** EMPTY SET *****&quot;</td>
</tr>
<tr>
<td><strong>RESULT-SET</strong></td>
<td></td>
<td></td>
<td>&quot;***** DYNAMIC SET *****&quot;</td>
</tr>
<tr>
<td><strong>SYS_INSTALLMENT</strong></td>
<td>SINGULAR</td>
<td></td>
<td>&quot;***** EMPTY SET *****&quot;</td>
</tr>
<tr>
<td><strong>SYS_ARTICLE-TYPE</strong></td>
<td>SINGULAR</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td><strong>SYS_ARTICLE-SELECTION</strong></td>
<td>SINGULAR</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td><strong>SYS_ARTICLE</strong></td>
<td>SINGULAR</td>
<td>63</td>
<td>63</td>
</tr>
<tr>
<td><strong>SYS_COLORS</strong></td>
<td>SINGULAR</td>
<td>25</td>
<td>25</td>
</tr>
<tr>
<td><strong>SYS_MATERIALS</strong></td>
<td>SINGULAR</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td><strong>IMPLICIT_RESULT_SET</strong></td>
<td>SINGULAR</td>
<td>2</td>
<td>8</td>
</tr>
</tbody>
</table>
BSTATUS does not print statistics for dynamic sets or for sets for which there are no set occurrences. Such sets are indicated as an *EMPTY SET*.

**SET IDENTIFICATION**
Name of set in schema

**OWNER OCCURRENCE**
Number of occurrences of the set
SINGULAR: SYSTEM set

**MEMBER OCCURRENCE**
Number of stored member records in the set
MIN: Number of member records in the smallest set occurrence
MAX: Number of member records in the largest set occurrence
AVG: Average number of member records per set

**TABLE STATISTICS**
Information on the tables in the set
(pointer arrays, sort key tables or SEARCH key tables)

If the set does not have any tables, the following message is printed:

**** CHAIN SET WITHOUT TABLES ****

**COL NR**
Column numbers in the owner DBTT containing the address of the given table

**FILLING RATIO**
Table occupancy level, expressed as a percentage of all bytes reserved for the table;
LEV=0: Average table occupancy on index level 0 (main level): Tables are set up, but no member records stored (empty set occurrence).
LEV>0: Average table occupancy on all index levels (main level not included)

**LEVEL NR**
Number of index levels in set (main level not included)
MAX: Highest index level in any table in a set occurrence
AVG: Average number of index levels in set

**NR OCCURRENCES TO REORGANIZE**
Number of set occurrences for which BREORG can be used to reduce the number of index levels; applies to any reorganization using the statement

REORGANIZE SET NAME IS set-name FILLING IS 100 PERCENT
Print owner statistics (DISPLAY TABLE FOR OWNER)

DISPLAY [IN CSV [csv-filename]] TABLE STATISTICS FOR OWNER IN SET

\[
\begin{aligned}
\{&set-name-1,\ldots, \\
&*ALL[\ EXCEPT \ set-name-1,\ldots]\}
\end{aligned}
\]

IN CSV

BSTATUS also outputs the data in CSV format.

csv-filename

Name of the file to which the data is to be output in CSV format. The specification of csv-filename is mandatory in the first IN CSV statement of a BSTATUS run (e.g. DISPLAY IN CSV 'BSTATUS.CSV' ...).

For a detailed description of CSV format output, see the manual “Database Operation”, section “Outputting database information in a neutral format”.

set-name

Name of a set for whose owner statistics are output

*ALL

Statistics are to be printed for all owner record types of the subschema

*ALL EXCEPT set-name-1,...

Statistics are to be printed for all sets other than set-name.

The DISPLAY TABLE FOR OWNER statement prints out statistics on the storage space occupied by the tables of the owner records of a set.

This output may be very large since, unlike the other statements, its scope does not depend only on the metadata population but also on the user data population.
Example

DISPLAY TABLE STATISTICS FOR OWNER IN SET PURCH-ORD-SPECS.
PURCH-ORD-SPECS

<table>
<thead>
<tr>
<th>OWNER DBK</th>
<th>OCCURRENCE</th>
<th>COL</th>
<th>FILLING RATIO</th>
<th>LEV</th>
<th>REORG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>NR</td>
<td>LEV=0</td>
<td>LEV&gt;0</td>
</tr>
<tr>
<td>-----------</td>
<td>------------</td>
<td>-----</td>
<td>-----</td>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>8:</td>
<td>1</td>
<td>7</td>
<td>1</td>
<td>47</td>
<td>0</td>
</tr>
<tr>
<td>8:</td>
<td>2</td>
<td>6</td>
<td>1</td>
<td>41</td>
<td>0</td>
</tr>
<tr>
<td>8:</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>14</td>
<td>0</td>
</tr>
<tr>
<td>8:</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>14</td>
<td>0</td>
</tr>
<tr>
<td>8:</td>
<td>11</td>
<td>7</td>
<td>1</td>
<td>47</td>
<td>0</td>
</tr>
<tr>
<td>8:</td>
<td>12</td>
<td>1</td>
<td>1</td>
<td>7</td>
<td>0</td>
</tr>
<tr>
<td>8:</td>
<td>13</td>
<td>1</td>
<td>1</td>
<td>7</td>
<td>0</td>
</tr>
</tbody>
</table>

BSTATUS prints a separate table for each set.

OWNER DBK
Database key of the owner record in the format: (recr:ref:rsq)

MEMBER OCCURRENCE
Number of members of the set occurrence specified by OWNER DBK
0: No member records stored; but table has been set up

TABLE STATISTICS
Information on all tables of the set occurrence
(pointer array, list or sort key table and SEARCH key tables)

COL NR
Column number in the owner DBTT containing the address of the corresponding table

FILLING RATIO
Table occupancy level, expressed as a percentage of all bytes reserved for the table;
DISPLAY RECORD statement

**BSTATUS**

LEV=0: Table occupancy level on index level 0 (main level):
0: Tables have been set up, but no member records have been stored
LEV>0: Table occupancy level on all index levels (excluding main level)

LEV NR
Number of index levels (excluding main level)

**REORG**
Indicates whether the number of index levels can be reduced by reorganization.
YES: the number of index stages can be reduced with the aid of the statement
REORGANIZE SETNAME IS set-name FILLING IS 100 PERCENT
in the BREORG utility
NO: it is not possible to reduce the number of index stages
#: the table has only one level (main level), so it is only possible to increase
table occupancy

**Print record type statistics (DISPLAY RECORD)**

DISPLAY [IN CSV [csv-filename]] RECORD STATISTICS FOR {record-name-1,...} ALL

**IN CSV**
BSTATUS also outputs the data in CSV format.

csv-filename
Name of the file to which the data is to be output in CSV format. The specification
of csv-filename is mandatory in the first IN CSV statement of a BSTATUS run (e.g.
DISPLAY IN CSV 'BSTATUS.CSV' ...).

For a detailed description of CSV format output, see the manual “Database
Operation”, section “Outputting database information in a neutral format”.

record-name
Name of a record type contained in the specified subschema for which record type
statistics are to be printed out

ALL  BSTATUS prints the record type statistics on all record types contained in the
specified subschema
**Example**

DISPLAY RECORD STATISTICS FOR ALL

**STATUS PER RECORD TYPE**

<table>
<thead>
<tr>
<th>RECORDS</th>
<th>NR OF ENTRIES USED</th>
<th>HIGHEST RSQ USED</th>
<th>HIGHEST RSQ POSSIBLE</th>
<th>FILLING RATIO %</th>
<th>EXTENDIBLE</th>
<th>NR OF DBTT EXTENTS</th>
<th>HIGHEST EXTENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUSTOMER</td>
<td>0</td>
<td>0</td>
<td>331</td>
<td>0,0</td>
<td>NO</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>CST-ORDERS</td>
<td>0</td>
<td>0</td>
<td>497</td>
<td>0,0</td>
<td>NO</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ORD-ITEM</td>
<td>0</td>
<td>0</td>
<td>1990</td>
<td>0,0</td>
<td>NO</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>INSTALMENT</td>
<td>0</td>
<td>0</td>
<td>995</td>
<td>0,0</td>
<td>NO</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ART-TYPE</td>
<td>4</td>
<td>4</td>
<td>497</td>
<td>0,8</td>
<td>NO</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ART-SELECTION</td>
<td>5</td>
<td>5</td>
<td>497</td>
<td>1,0</td>
<td>NO</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ART-DESCR</td>
<td>13</td>
<td>13</td>
<td>497</td>
<td>2,6</td>
<td>NO</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ARTICLE</td>
<td>63</td>
<td>63</td>
<td>995</td>
<td>6,3</td>
<td>NO</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>P-ORD-ITEM</td>
<td>0</td>
<td>0</td>
<td>995</td>
<td>0,0</td>
<td>NO</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>TOTAL</strong></td>
<td><strong>121</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**RECORDS**

Names of record types

**DBTT**

For each record type BSTATUS prints out the most important DBTT information

**NR OF ENTRIES USED**

The number of DBTT entries used is equivalent to the number of currently stored records of the record type

**HIGHEST RSQ USED**

Highest record sequence number assigned; if records have been erased, this number may be higher than the number of records currently stored

**HIGHEST RSQ POSSIBLE**

Highest record sequence number possible, i.e. greatest number of records of this record type which can be stored
FILLING RATIO %
DBTT occupancy level, expressed as a percentage of the allocated DBTT entries

EXTENDIBLE:
Specifies whether or not the record type can be extended:

- NO: cannot be extended
- YES: can be extended with parameter SCAN=YES
- NOSCAN: can be selected with parameter SCAN=NO

NR OF DBTT EXTENTS:
Number of DBTT extents currently present

HIGHEST USED EXTENT:
Number of DBTT extents currently present minus the number of consecutive completely empty extents at the end of the DBTT. This information can be of use in the case of a DBTT reduction using BREORG.

TOTAL
Outputs the total number of DBTT entries used.
Print CALC key statistics (DISPLAY CALC)

DISPLAY [IN CSV [csv-filename]] CALC KEY STATISTICS FOR

{ RECORD {record-name-1,...} ALL }
{ SEARCHKEY {keyref-1,...} ALL }
{ IN REALM {realmname-1,...} ALL }

IN CSV

BSTATUS also outputs the data in CSV format.

`csv-filename`
Name of the file to which the data is to be output in CSV format. The specification of `csv-filename` is mandatory in the first IN CSV statement of a BSTATUS run (e.g. `DISPLAY IN CSV 'BSTATUS.CSV' ...`).

For a detailed description of CSV format output, see the manual “Database Operation”, section “Outputting database information in a neutral format”.

RECORD

BSTATUS prints CALC key statistics for record types defined with LOCATION MODE IS CALC

`record-name`
Name of a record type for which BSTATUS is to print out the CALC key statistics

ALL
BSTATUS prints CALC key statistics for all record types which have been defined with LOCATION MODE IS CALC in the specified subschema

SEARCHKEY

BSTATUS prints the CALC key statistics for the hash areas of CALC SEARCH keys defined on record type or set level

`keyref`
Number of the CALC SEARCH key whose CALC key statistics are to be printed by BSTATUS.
The numbers can be taken from the SIA PRINT REPORT (see page 142)

ALL
BSTATUS prints CALC key statistics for all hash areas of CALC SEARCH keys that are contained in the subschema
REALM
BSTATUS prints CALC key statistics for one or more realms

realmname
Name of a realm for which BSTATUS is to print out the CALC key statistics

ALL
BSTATUS prints CALC key statistics for all realms

Example
DISPLAY CALC SEARCHKEY 12,14,15,16,17 IN REALM ARTICLE-RLM.

STATUS OF CALC KEY BUCKETS - PER CALC KEY
REALM: ARTICLE-RLM

<table>
<thead>
<tr>
<th>CALC KEYS</th>
<th>NR OF PRIMARY BUCKETS</th>
<th>RECORDS/ POINTERS THAT CAN BE ADDED</th>
<th>NR OF EMPTY BUCKETS</th>
<th>FILLING % PRIMARY BUCKETS</th>
<th>NR OF OVERFLOW BUCKETS</th>
<th>RECORDS/ POINTERS IN OVERFLOW</th>
<th>FILLING % OVERFLOW BUCKETS</th>
<th>DEPTH FACTOR FOR RECORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEY-REF: 12</td>
<td>1</td>
<td>109</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>KEY-REF: 14</td>
<td>3</td>
<td>597</td>
<td>0</td>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>KEY-REF: 15</td>
<td>11</td>
<td>806</td>
<td>3</td>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>KEY-REF: 16</td>
<td>1</td>
<td>107</td>
<td>0</td>
<td>19</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>KEY-REF: 17</td>
<td>1</td>
<td>305</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-</td>
</tr>
</tbody>
</table>

BSTATUS prints separate CALC key statistics for each realm.

RECORDS
Only if DISPLAY CALC RECORD is specified:
Names of the record types for which BSTATUS is to print out the CALC key statistics

CALC KEYS
KEY REF: keyref
Only if DISPLAY CALC SEARCHKEY is specified:
Numbers of CALC SEARCH keys for which BSTATUS is to print out the CALC key statistics

NR OF PRIMARY BUCKETS
Number of primary pages reserved for the hash area
RECORDS/POINTERS THAT CAN BE ADDED
Number of records (for a direct hash area) or pointers (for an indirect hash area) that can still be stored in the primary pages of the hash area

NR OF EMPTY BUCKETS
Number of empty primary pages

FILLING % PRIMARY BUCKETS
Occupancy level of the primary pages

NR OF OVERFLOW BUCKETS
Current number of overflow pages

RECORDS/POINTERS IN OVERFLOW
Number of records (for a direct hash area) or number of pointers (for an indirect hash area) in the overflow pages

FILLING % OVERFLOW BUCKETS
Occupancy level i.e. occupancy level of the overflow pages

DEPTH FACTOR FOR RECORD
Depth factor for accessing a record, i.e. average number of accesses required for locating a record

BSTATUS calculates the depth factor according to the following formulas:

– Direct hash area

\[
d = \frac{\sum_{i=1}^{n} (\text{no. of records in page}_i \times \text{rank}_i)}{\text{total no. of records}}
\]

– Indirect hash area:

\[
d = \frac{\sum_{i=1}^{n} (\text{no. of pointers in page}_i \times \text{rank}_i)}{\text{total no. of pointers}} + 1
\]

\[n\] Number of all pages in the hash area (primary and overflow pages)

\[rank_i\] Ranking of the i-th CALC page in a chain of primary and overflow pages

Primary page: rank 1
1st overflow page: rank 2
2nd overflow page: rank 3
and so forth
$d$  Depth factor

*total no. of records*
Number of all records stored in the primary and overflow pages of the hash area

*total no. of pointers*
Number of all pointers stored in the primary and overflow pages of the hash area
Print record number statistics (DISPLAY RECORDNUMBER)

DISPLAY [IN CSV [csv-filename]] RECORDNUMBER STATISTICS FOR

\{
\text{REALM} \{ \text{realm-name-1,...} \} \\
\text{ALL} \\
\text{RECORD} \{ \text{record-name-1,...} \} \\
\text{ALL} \\
\}\}

IN CSV

BSTATUS also outputs the data in CSV format.

csv-filename

Name of the file to which the data is to be output in CSV format. The specification of csv-filename is mandatory in the first IN CSV statement of a BSTATUS run (e.g. DISPLAY IN CSV 'BSTATUS.CSV' ...).

For a detailed description of CSV format output, see the manual “Database Operation”, section “Outputting database information in a neutral format”.

REALM

For each record type which can occur in the specified realm(s), BSTATUS prints the number of records stored therein

realm-name

Name of the realm for which BSTATUS is to print out the record number statistics

ALL

BSTATUS prints the record number statistics for all realms of the subschema

RECORD

For each specified record type, BSTATUS prints out the number of records stored in the realms in which the record type can occur

record-name

Name of the record type for which BSTATUS is to print out the record number statistics

ALL

BSTATUS prints the record number statistics for all record types of the subschema
### Example

DISPLAY RECORDNUMBER STATISTICS FOR REALM ALL.

#### NUMBER OF RECORDS PER REALM

<table>
<thead>
<tr>
<th>REALM REFS</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUSTOMER</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CST-ORDERS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ORD-ITEM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>INSTALLMENT</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>ART-TYPE</td>
<td></td>
<td></td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ART-SELECTION</td>
<td></td>
<td></td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ART-DESCR</td>
<td></td>
<td></td>
<td>8</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ARTICLE</td>
<td></td>
<td></td>
<td>55</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SUBSET</td>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td>0</td>
<td>-</td>
<td>0</td>
<td>-</td>
</tr>
<tr>
<td>COLORS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>MATERIALS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>SUPPLIER</td>
<td></td>
<td></td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PURCHASE-ORDER</td>
<td></td>
<td></td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>P-ORD-ITEM</td>
<td></td>
<td></td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AREAS REF</th>
<th>3 = CUSTOMER-ORDER-RLM</th>
</tr>
</thead>
<tbody>
<tr>
<td>AREA REF 4</td>
<td>4 = PURCHASE-ORDER-RLM</td>
</tr>
<tr>
<td>AREA REF 5</td>
<td>5 = CLOTHING</td>
</tr>
<tr>
<td>AREA REF 6</td>
<td>6 = HOUSEHOLD-GOODS</td>
</tr>
<tr>
<td>AREA REF 7</td>
<td>7 = SPORTS-ARTICLES</td>
</tr>
<tr>
<td>AREA REF 8</td>
<td>8 = FOOD</td>
</tr>
<tr>
<td>AREA REF 9</td>
<td>9 = LEISURE</td>
</tr>
<tr>
<td>AREA REF 10</td>
<td>10 = STATIONERY</td>
</tr>
</tbody>
</table>
RECORDS
Names of record types

REALM REFS
Realm numbers; for each realm BSTATUS sets up a column containing the number of records stored in the realm
0: The record type can be contained in the realm, but no record is stored there
-: The record type cannot occur in the realm

AREA REF
Assignment of realm numbers to realm names

Terminate input (END)

END
6.4 Command sequence to start BSTATUS

The command sequence described here is based on the assumption that UDS/SQL was installed with IMON (see the section “START commands of the UDS/SQL programs” in chapter 2 of the “Creation and Restructuring” manual).

01 /ADD-FILE-LINK LINK-NAME=DATABASE,
   FILE-NAME=[:catid:][(userid.dbname.DBDIR[.copy-name]]

02 /SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL,VERSION=version.

03 /START-UDS-BSTATUS

04 SUBSCHEMA IS subschema-name

05 display-statements

06 END

01 In this case specifying :catid: is permitted (see in the “Database Operation” manual).

02 The version of the utility routine is selected.
Specification of the version is generally recommended, since several UDS/SQL versions can be installed in parallel.

03 The UDS/SQL utility routine can also be started with the alias BSTATUS.
7 Printing out the contents of realms with BP RECORD

BP RECORD prints the contents of the user's database.

The following information can be obtained from the printout:
- from the realm information page (Act-Key-0 page) the date of the last realm update and system break information
- from the FPA entries the location and size of unused space in the realms (important for reorganizing a realm)
- from the DBTTs the available record sequence numbers (important for LOCATION MODE IS DIRECT)
- from the printout of the hash areas the distribution of records over the hash area (basis for improving hash routines)
- from the printout of data records and tables the connection of records to tables (useful for debugging DB applications)

With BP RECORD the DB administrator can print the contents of the user realms, of the database directory (DBDIR) or of the database compiler realm (DBCOM).

The DB administrator can select individual sections of a realm according to two aspects:

Logical  The administrator selects a particular record type, certain records of a record type, a table, etc.

Physical  The administrator selects the realm, certain page types certain page numbers, etc.

The two options may also be combined. When printing out CALC and data pages, the administrator may also specify whether BP RECORD is to include the page header (PAGE INFO), the page index entries (PAGE INDEX) or the set connection data (SCD) of the records in the BP RECORD printout (see page 217).

To prevent unauthorized access, BP RECORD can only be invoked under the DB administrator identification.
In addition to outputting the data to SYSLST, you can also output it to a file in CSV format. The use of CSV format facilitates the further processing of the data in other system environments (e.g. in spreadsheet applications). The output in CSV format is described in the manual “Database Operation”, section “Outputting database information in a neutral format”.

When you run BPRECORD online, you must assume that the data output is not current since the DBH has not copied all the data from the buffer to the database yet. In order to obtain as much current data as possible, you should force a database update just before the BPRECORD run using the DAL command CHECKPOINT or NEW RLOG. However, the BPRECORD output may still differ from the actual contents of the database if an update is running parallel to this task.
7.1 System environment

Figure 12: BPRECORD system environment
7.2 General description of the output of BPRECORD

BPRECORD is used to print out any combination of the five different page types making up a realm in the following sequence:

1.  
   ************************************* ACTUAL-KEY-0 OF REALM < :H32:$XXXXXXXX.SHIPPING.ARTICLE-RLM > ************* 
   PHYS REC LENGTH : 4000 
   DATASET INFO : 
   TOTAL NR PAGES: 2064 
   FPA BASE BEGIN: 11 
   AREA REF: 11 
   BNR: 1 
   NR PAGES IN FPA BASE: 1990 
   FPA EXTENTS 
<table>
<thead>
<tr>
<th>NR</th>
<th>BNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>47</td>
</tr>
</tbody>
</table>
   MAX NR PAGES IN EXTENT: 63680 
   HIGHEST PAGE NR FOR FORMATTING: 2063 
   CREATE DATA: DATE : 20040812 TIME: 154031 
   BACK UP DATA: DATE : 20040812 TIME: 154118 
   REALM VERSION NR: 3 
   SYSTEM BREAK: 
   OCCURRED: 0 
   ADMIN USERID: $XXXXXXXX 
   CONFNAME: BREORG 
   FILE NAME: :H32:$XXXXXXXX.SHIPPING.ARTICLE-RLM 
   REALM LAYOUT VERSION: 004.00 
   UDS VERSION: V2.4 
   INCR NR PAGES: 0 
   INCR MIN PAGES: 0 

2.  
   ************************************* F.P.A. ENTRIES OF REALM < :H32:$XXXXXXXX.SHIPPING.ARTICLE-RLM > ************* 
   -------------------------------------- REALM REF 11 BNR 47 ACT KEY X'0B000001' ------------------------------------- 
<table>
<thead>
<tr>
<th>REALM REF</th>
<th>BNR</th>
<th>ACT KEY</th>
</tr>
</thead>
<tbody>
<tr>
<td>0..29</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>30.39</td>
<td>0</td>
<td>3960</td>
</tr>
<tr>
<td>40.49</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>50.69</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>70.79</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>80.99</td>
<td>3980</td>
<td>3980</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
==================================== RECORD REF 6, NAME: < ART-TYPE >=====================================
-------------------------------------- REALM REF 11 BNR 3 ----- ACT KEY X'OB00003' -------------------------------------

RSQ 1/X'00000001': ( 5, 12/X'05000000C') ( 11, 37/X'OB000025')
RSQ 2/X'00000002': ( 8, 4/X'OB000025') ( 11, 39/X'OB000027')
RSQ 3/X'00000003': ( 8, 5/X'OB000025') ( 11, 41/X'OB000029')
RSQ 4/X'00000004': ( 8, 6/X'OB000025') ( 11, 44/X'OB00002C')
RSQ 5/X'00000005': ( 0, 0/X'00000000') ( 0, 0/X'00000000')

4. *********************************** CALC KEY BUCKETS OF REALM < :H32:$XXXXXXXX.SHIPPING.ARTICLE-RLM > ***********
==================================== CALC SEARCH KEY, KEY REF 14, SET NAME < SYS_ARTICLE >=====================================
-------------------------------------- REALM REF 11 BNR 16 ----- ACT KEY X'OB000010' -------------------------------------

PAGE INFO: TYPE 0 — FREE SPACE SIZE 10, DISPL 30 — NR OF PAGE INDICES 0 — DISPL TO CALC TABLE HEADER 30
CALC KEY TABLE: MAX ENTRIES 220 — ACT ENTRIES 25 — OVERFLOW BUCKET NEXT 0, PRIOR 0

-1-
( 1) (00000000) F2F3F0F1 FDF7F3F6 23010736
RSQ 1 — PPP 5, 13
-2-
( 1) (00000000) F2F3F0F1 FDF7F4F2 23010742
RSQ 4 — PPP 5, 13
-3-
( 1) (00000000) F2F3F0F1 FDF7F4F8 23010748
RSQ 7 — PPP 5, 13
-4-

5. ********************************** DATA / TABLE PAGES OF REALM < :H32:$XXXXXXXX.SHIPPING.CLOTHING   > **********
==================================== REALM REF 5 BNR 19 ----- ACT KEY X'05000013'=====================================

PAGE INFO: TYPE 0 — FREE SPACE SIZE 2081, DISPL 2209 — NR OF PAGE INDICES 9 — DISPL TO END OF PAGE 4000
LOGICAL RECORDS:
-2-
PAGE INDEX: DB_KEY 9, 48 — COL-NR 0 — LIST REC — DISPL 3767
( 1) (000000000) 00090000 00000030 05000013 00090000 00000030 05000013 00090000 00000030
( 33) (000000000) 05000013 00090000 00000030 05000013 00090000 00000030 05000013 00090000
( 65) (000000000) 00000030 05000013 00090000 00000030 05000013 00090000 00000000 000A0500 0012FF00
( 97) (000000000) 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0400000E
(129) (000000080) FBF3F5F9 F2F8F0F9 E360E2C8 C9D9E340 C4D9C5E2 E240400C 0023000C 0023000C 23710236
(161) (0000000A0) 40404040 40404040 40404040 40404040 F2F3F7F1 FDF7F3F6 0020700C 0023900C 23710236

The structure and function of the different page types are described in the “Design and Definition” manual.

SCHEMA NAME schema-name
Name of the schema, whose realms are to be printed by BPRECORD
ACTUAL-KEY-0 OF REALM realm-name
Realm information page of the realm realm-name; BPRECORD optionally prints the realm information page in normal text

F.P.A. ENTRIES OF REALM realm-name
FPA (free place administration) entries of the realm realm-name; BPRECORD optionally prints all FPA entries or only the FPA entries of certain pages

REALM REF realm-ref BNR pnr
BPRECORD prints the page address before the contents of each page
realm-ref: Number of the realm
pnr: Page number

D.B.T.T. ENTRIES OF REALM realm-name
DBTT entries of the realm realm-name; BPRECORD optionally prints:
– all DBTTs,
– only the DBTT of a certain record type, or
– only the DBTT entries of certain records of a record type

RECORD REF recref, NAME: record-name
is the header of the DBTT of a record type
recref: Number of the record type (record reference)
record-name: Name of the record type

CALC KEY BUCKETS OF REALM realm-name
Direct CALC pages or indirect CALC pages (primary and overflow pages) of the realm realm-name

CALC RECORD, REC REF recref, RECORD NAME record-name
Direct or indirect CALC pages of the record type record-name

CALC SEARCH KEY, KEY REF keyref, SET NAME set-name
CALC pages of CALC SEARCH key keyref
keyref: Number of the key
set-name: Name of the associated set

DATA/TABLE PAGES OF REALM realm-name
Data pages of the realm realm-name, which contain records (excluding CALC records) or tables and/or table entries;

BPRECORD only prints pages which are not empty. To decide whether or not a page is empty, BPRECORD first checks the FPA entries.
CALC pages and data pages are sorted according to ascending page numbers.
### 7.3 BPRECORD statements

<table>
<thead>
<tr>
<th>Statement</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCHEMA NAME</td>
<td>Name of schema which contains the realm to be printed (optional)</td>
</tr>
<tr>
<td>REALM NAME</td>
<td>Name of the realm to be printed (mandatory)</td>
</tr>
<tr>
<td>PRINT</td>
<td>Determine extent of output (optional)</td>
</tr>
<tr>
<td>DISPLAY PAGE</td>
<td>Print Act-Key-0 page (optional)</td>
</tr>
<tr>
<td>DISPLAY FPA</td>
<td>List FPA entries (optional)</td>
</tr>
<tr>
<td>DISPLAY DBTT</td>
<td>List DBTT entries (optional)</td>
</tr>
<tr>
<td>DISPLAY CALC</td>
<td>Print CALC pages (optional)</td>
</tr>
<tr>
<td>DISPLAY DATA</td>
<td>Print data pages (optional)</td>
</tr>
<tr>
<td>END</td>
<td>Terminate BPRECORD run (mandatory)</td>
</tr>
</tbody>
</table>

Table 17: BPRECORD statements

BPRECORD combines all DISPLAY statements for a certain realm and sorts them to avoid duplicate printouts.

For purposes of clarity, PRINT and DISPLAY statements are best separated by a semicolon.

If a period (.) were used for separation, this would signify the end of statements entered for one realm. BPRECORD would then expect a new REALM statement or the END statement to be entered.

After SCHEMA-NAME/REALM-NAME, you must specify at least one DISPLAY statement.

**Example for a statement sequence**

```plaintext
SCHEMA NAME IS schema-name.
REALM NAME IS realm-name-1.
PRINT WITH SCG;
DISPLAY DATA PAGES ALL PAGES ALL TABLES;
DISPLAY DBTT OF ALL RECORDS.
REALM NAME IS realm-name-2.
PRINT WITH PAGEINDEX;
DISPLAY FPA OF ALL PAGES;
DISPLAY CALC PAGES ALL PAGES ALL RECORDS.
END
```
Physical selection (page selection)

The syntax elements `page-selection` and `rsq-selection` are used in a number of different statements:

\[
\text{page-selection} := \begin{cases} 
\text{ALL PAGES} \\
\text{PAGE \{pno-1[ TO pno-2]\},...} 
\end{cases}
\]

ALL PAGES

The total collection of pages defined by the logical selection

PAGE `pno-1`,...
List of page numbers

PAGE `{pno-1 TO pno-2}`,...
Range from page number `pno-1` to page number `pno-2`, etc.

Logical selection (RSQ selection)

\[
\text{rsq-selection} := \begin{cases} 
\text{ALL RSQS} \\
\text{RSQ \{rsq-1[ TO rsq-2]\},...} 
\end{cases}
\]

ALL RSQS

All record sequence numbers

RSQ `rsq-1`,...
List of record sequence numbers

RSQ `{rsq-1 TO rsq-2}`,...
Range of record sequence numbers from `rsq-1` to `rsq-2`, etc.
Designate the schema (SCHEMA NAME)

```
SCHEMA NAME IS schema-name.
```

**schema-name**

Name of the schema containing the description of the realm or realms to be printed;  
The following can be specified for *schema-name*:

- *user-schema-name* for a printout of a user realm  
- COMPILER-SCHEMA for a printout of the DBCOM  
- PRIVACY-AND-IQF-SCHEMA for a printout of the DBDIR

Default value:  
User schema

The SCHEMA statement is optional. If it is specified, it must be the first BPRECORD statement, and may only be entered once.

BPRECORD accesses the SIA of the specified schema and obtains from it all information required to access the database.
REALM NAME statement

Specify the realm to be printed (REALM NAME)

REALM NAME IS realm-name.

realm-name
   Name of the realm to be printed; realm-name must be specified as follows:
   – for user realms: the realm name defined in the AREA clause of the Schema DDL
   – for the DBDIR: DATABASE DIRECTORY
   – for the DBCOM: DATABASE COMPILER REALM

The REALM statement must be entered at least once; it may be repeated any number of times. Note that all PRINT and DISPLAY statements referring to a particular realm must immediately follow the corresponding REALM statement. The first REALM statement must immediately follow the SCHEMA statement, or, if there is none, must be entered as the first BPRECORD statement (see page 211).
Determine scope of output (PRINT)

\[
\text{PRINT[} \begin{cases} \text{WITH} \\ \text{WITHOUT} \end{cases} \text{ PAGEINFO[} \begin{cases} \text{WITH} \\ \text{WITHOUT} \end{cases} \text{ PAGEINDEX} \\
\begin{cases} \text{WITH} \\ \text{WITHOUT} \end{cases} \text{ SCD[} \begin{cases} \text{DEC} \\ \text{HEX} \end{cases} \text{ DBTT} \begin{cases} \text{DEC} \\ \text{HEX} \end{cases} \end{cases}]:
\]

PAGEINFO
Page header

PAGEINDEX
Page index entries

SCD  Set connection data

PAGEINFO, PAGEINDEX and SCD apply to the output of CALC and data pages.

DEC  Decimal
HEX  Hexadecimal
BOTH Decimal and hexadecimal

DEC, HEX and BOTH apply to the output of DBTTs.

Default values:
\begin{cases} \text{WITHOUT} \\ \text{DEC} \end{cases}

Use of the PRINT statement is optional. It affects all DISPLAY statements which the user enters between two REALM statements or between a REALM statement and the END statement.
If the PRINT statement appears more than once, only the last statement takes effect.
Example

PRINT WITH PAGEINFO WITH PAGEINDEX WITH SCD.

************************************************************************ DATA / TABLE PAGES OF REALM < :H32:$XXXXXXX.SHIPPING.CLOTHING > **********

REALM REF  5  BNR   18  ACT KEY '05000012'

PAGE INFO:  TYPE 0 -- FREE SPACE  SIZE 1963, DISPL 2103 -- NR OF PAGE INDICES 10 -- DISPL TO END OF PAGE 4000

LOGICAL RECORDS:

-2- PAGE INDEX: DB_KEY 9, 41 -- COL-NR 0 -- LIST_REC -- DISPL 3767
(  1) (00000000) 0090000 0000029 05000012 0090000 0000029 05000012 0090000 0000029
( 33) (00000020) 05000012 0090000 0000029 05000012 0090000 0000029 05000012 0090000
( 65) (00000040) 0000029 05000012 0090000 0000029 05000012 0090000 0000029 05000012
( 97) (00000060) 0000000 0000000 0000000 0000000 0000000 0000000 0000000 0000000
(129) (00000080) F8F3F1F2 F1F3F1F1 C6D3D6E6 C9D5C740 D1C5D9E2 C5E840C4 D9C5E2E2 40E6C9E3
(161) (000000A0) C840D1C1 C3D2C5E3 40404040 40404040 F2F3F6F1 F1F3F6F1 0020700C
(193) (000000C0) 00000500 00000500 00000499 950C0000 00000000 00000000

-3- PAGE INDEX: DB_KEY 9, 42 -- COL-NR 0 -- LIST_REC -- DISPL 3552
(  1) (00000000) 0090000 000002A 05000012 0090000 000002A 05000012 0090000 000002A
( 33) (00000020) 05000012 0090000 000002A 05000012 0090000 000002A 05000012 0090000
( 65) (00000040) 000002A 05000012 0090000 000002A 05000012 0090000 000002A 05000012
( 97) (00000060) 0000000 0000000 0000000 0000000 0000000 0000000 0000000 0000000
(129) (00000080) F8F3F1F2 F1F3F1F1 C6D3D6E6 C9D5C740 D1C5D9E2 C5E840C4 D9C5E2E2 40E6C9E3
(161) (000000A0) C840D1C1 C3D2C5E3 40404040 40404040 F2F3F6F1 F1F3F6F1 0020700C
(193) (000000C0) 00000500 00000500 00000499 950C0000 00000000 00000000

-4- PAGE INDEX: DB_KEY 9, 43 -- COL-NR 0 -- LIST_REC -- DISPL 3337
(  1) (00000000) 0090000 000002B 05000012 0090000 000002B 05000012 0090000 000002B
( 33) (00000020) 05000012 0090000 000002B 05000012 0090000 000002B 05000012 0090000
( 65) (00000040) 000002B 05000012 0090000 000002B 05000012 0090000 000002B 05000012
( 97) (00000060) 0000000 0000000 0000000 0000000 0000000 0000000 0000000 0000000
(129) (00000080) F8F3F1F2 F1F3F1F1 C6D3D6E6 C9D5C740 D1C5D9E2 C5E840C4 D9C5E2E2 40E6C9E3
(161) (000000A0) C840D1C1 C3D2C5E3 40404040 40404040 F2F3F6F1 F1F3F6F1 0020700C
(193) (000000C0) 00000500 00000500 00000499 950C0000 00000000 00000000

...
PAGE INFO (page header)

All pages, except for DBTT pages and FPA pages, contain the page header in the first 20 bytes. BPRECORD prints the following PAGE INFO:

**TYPE**  Type of page
0:  Data page or CALC page
1:  Act-Key-0 page

**FREE SPACE**
Free storage space in the page:

**SIZE**
Length of the free storage space (in bytes)

**DISPL**
Displacement relative to the first unused byte

**NO OF PAGE INDICES**
Number of page index entries

**DISPL TO END OF PAGE**
Length of the page (in bytes)

PAGE INDEX (page index entry)

Page index entries are used to locate a record or table within the page. They occur in data pages and direct CALC pages. BPRECORD prints:

**DB-KEY**
Database key of a record in the form: *recref*, *rsq*

**COL-NR**
Column number in the DBTT
=0:  Data record (LOGICAL RECORD or CALC KEY REC)
>0:  Table record (TABLE REC)

**DISPL**  Displacement relative to the data record

Set connection data

The set connection data is explained in the “*Design and Definition*” manual.
Print act-key-0 page (DISPLAY PAGE)

```plaintext
DISPLAY [IN CSV [csv-filename]] PAGE ZERO;
```

The DISPLAY-PAGE-ZERO statement prints the first 108 bytes of the realm information page (Act-Key-0 page) in normal text. Use of this statement is optional; repeated input for the same realm is ignored by BPRECORD.

**IN CSV**

BPRECORD also outputs the data in CSV format.

*csv-filename*  
Name of the file to which the data is to be output in CSV format. The specification of *csv-filename* is mandatory in the first IN CSV statement of a BPRECORD run (e.g. DISPLAY IN CSV 'BPRECORD.CSV' ...).

For a detailed description of CSV format output, see the manual “Database Operation”, section “Outputting database information in a neutral format”.

**Example**

```plaintext
DISPLAY PAGE ZERO
```

```
*********************************************************************** ACTUAL-KEY-0 OF REALM < :H32:$XXXXXXXX.SHIPPING.ARTICLE-RLM > *************
PHYS REC LENGTH :        4000
DATASET INFO :  TOTAL NR PAGES:                 2064
                 FPA BASE          BEGIN:        AREA REF:           11
                 BNR:                 1
                 NR PAGES IN FPA BASE:           1990
FPA EXTENTS
   NR           BNR
   1            47
   MAX NR PAGES IN EXTENT:                   63680
HIGHEST PAGE NR FOR FORMATTING:                   2063
CREATE DATA:      DATE :        20040812   TIME:      154031
BACK UP DATA:     DATE :        20040812   TIME:      154118
REALM VERSION NR:           3
SYSTEM BREAK:
   OCCURRED:     0
   ADMIN USERID: $XXXXXXXX
   CONFNAME:     BREORG
FILE NAME:            :H32:$XXXXXXXX.SHIPPING.ARTICLE-RLM
REALM LAYOUT VERSION:           004.00
UDS VERSION:                    V2.4
INCR NR PAGES:                            0
INCR MIN PAGES:                           0
```
PHYS REC LENGTH
Page length (in bytes)

DATASET INFO
General information on the realm:

FPA BASE
Information on the FPA base

AREA REF
Number of the realm that this free place administration applies to in the FPA base.

BNR
Page number of the first FPA page

TOTAL NR PAGES
Total number of pages administered in the realm

NR PAGES FPA BASE
Number of pages past the FPA base administered in the realm

FPA EXTENTS
Information on all FPA extents of the realm

NR
Number of FPA extent

BNR
Number of the first page of the FPA extent

MAX NR PAGES IN EXTENT
Maximum number of data pages administered in a FPA extent

HIGHEST PAGE NR FOR FORMATTING
Number of the database page up to which a format applies when extending a realm

nnnnnnnn
Number of the database page up to which a format applies when extending a realm.

If the value is the same as the value of TOTAL NR PAGES, then any new pages added to a realm during a realm extension are not formatted.

If the value is greater than the value of TOTAL NR PAGES, then all new pages up to the specified number that are added to a realm during a realm extension are formatted.

In the case of 2-Kbyte databases, further conditions must be fulfilled for reasons of safety thus making it unnecessary to format newly added pages.
UNKNOWN
All pages added to a realm during a realm extension are formatted.

CREATE DATA
Creation date for the realm:

DATE
Creation date in the format: yyyymmdd

TIME
Creation time in the format: hhmms

BACK UP DATA
Time last update occurred see CREATE DATA for format)

DATE
Date of last update

TIME
Time of last updates

REALM VERSION NR
Internal version number of the realm; only changed by utility routines intended for updates (e.g. BALTER, BCHANGE, BOUTLOAD, BREORG)

SYSTEM BREAK
Indicates whether or not the realm was closed correctly

OCCURRED
0: Last session terminated normally
1: Last session aborted, i.e. the realm may be inconsistent

ADMIN USERID
ID under which the database is maintained.

CONFFNAME
Name of configuration or utility routine with which the database was last accessed

FILE NAME
Complete file name of the realm

DATABASE LAYOUT VERSION
Version number of present database layout version in the format nnn.nn (output for the DBDIR)

REALM LAYOUT VERSION
Version number of the present realm layout structure in the format nnn.nn

UDS VERSION
UDS/SQL version number under which the database was last modified by DBH. The field is used to ensure that warm starts are performed correctly.
INCR
Information on the online extensibility of the realm. This information is output for the
DBDIR and the user realms.
The information is only output for realms for which the online extensibility was acti-
vated with the DAL command ACT INCR.
The information is also output when the online extensibility is deactivated.

NR PAGES
Number of pages added to the realm during an online extension.

MIN PAGES
Minimum number of free pages. If the number of free pages drops below
this value, then an online extension is triggered by the DBH for the corre-
sponding realm.
List FPA entries (DISPLAY FPA)

```
DISPLAY [IN CSV [csv-filename]] FPA OF page-selection;
```

IN CSV

BPRECORD also outputs the data in CSV format.

*csv-filename*

Name of the file to which the data is to be output in CSV format. The specification of *csv-filename* is mandatory in the first IN CSV statement of a BPRECORD run (e.g. DISPLAY IN CSV 'BPRECORD.CSV' ...).

For a detailed description of CSV format output, see the manual “Database Operation”, section “Outputting database information in a neutral format”.

*page-selection*

See “Physical selection (page selection)” on page 212

The DISPLAY FPA statement prints the FPA entries of a realm as a whole or in part. Use of this statement is optional. If you specify several FPA statements, BPRECORD combines the statements internally to a single input.

**Example**

```
DISPLAY FPA OF ...
```

```
0........29: 0 0 0 0 0 0 0 0 0 0
30........39: 0 0 3960 3964 0 0 3956 0 0 0
40........49: 0 0 0 0 0 0 0 0 0 0
50........69: 0 0 0 0 0 0 0 0 0 0
70.......79: 0 0 0 0 0 0 0 0 0 3980
80.....1989: 3980 3980 3980 3980 3980 3980 3980 3980 3980 3980
```

```
1990.....2059: 3980 3980 3980 3980 3980 3980 3980 3980 3980 3980
2060.....2063: 3980 3980 3980 0
```

<table>
<thead>
<tr>
<th>page numbers</th>
<th>free bytes per page</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>page full</td>
</tr>
<tr>
<td>1 ... 2027</td>
<td>page partly filled</td>
</tr>
<tr>
<td>2028</td>
<td>page empty</td>
</tr>
</tbody>
</table>

Invalid FPA values are marked XX.
List DBTT entries (DISPLAY DBTT)

DISPLAY [IN CSV [csv-filename]] DBTT \{ALL RECORDS
  \{RECORD record-name FOR rsq-selection\}\};

IN CSV
BPRECORD also outputs the data in CSV format.

csv-filename
Name of the file to which the data is to be output in CSV format. The specification of csv-filename is mandatory in the first IN CSV statement of a BPRECORD run (e.g. DISPLAY IN CSV 'BPRECORD.CSV' ...).

For a detailed description of CSV format output, see the manual “Database Operation”, section “Outputting database information in a neutral format”.

ALL-RECORDS
Prints the DBTTs of all record types contained in the specified realm.

record-name
Name of a record type whose DBTT (or DBTT entries) is/are to be printed.

rsq-selection
See “Logical selection (RSQ selection)” on page 212.

The DISPLAY DBTT statement is optional and can be repeated. Repeated input of the same DISPLAY DBTT statement is ignored by BPRECORD.

If multiple page numbers are specified in the DISPLAY DBTT statement, they must be specified in ascending order.

DBTT entries can be listed in decimal and/or hexadecimal format (see the PRINT statement).
**Example**

DISPLAY DBTT OF RECORD.ARTICLE-TYPE FOR RSQ 1 TO 11 (with PRINT DBTT BOTH)

*************************************************** D.B.T.T. ENTRIES OF REALM < :H32:$XXXXXXXX.SHIPPING.ARTICLE-RLM > ********************

---

<table>
<thead>
<tr>
<th>RSQ</th>
<th>REALM REF</th>
<th>BNR</th>
<th>ACT KEY</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/X'00000001':</td>
<td>11</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>2/X'00000002':</td>
<td>11</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>3/X'00000003':</td>
<td>11</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4/X'00000004':</td>
<td>11</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>5/X'00000005':</td>
<td>11</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>6/X'00000006':</td>
<td>11</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>7/X'00000007':</td>
<td>11</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>8/X'00000008':</td>
<td>11</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

![Diagram of DBTT entries]

**Notes:**
- The diagram shows the layout of the DBTT entries, with the decimal and hexadecimal values for each column.
- The entries are formatted to show the sequence number, realm number, page number, and the address of the data record and set occurrence table.

---

**BPRECORD**
**Print CALC pages (DISPLAY CALC)**

```plaintext
display [in csv [csv-filename]] calc pages page-selection
{
  all [records
      calc searchkeys
    ]

  only {record record-name
       calc searchkey keyref
    } rsq-selection
}
```

**IN CSV**

BPRECORD also outputs the data in CSV format.

*csv-filename*

Name of the file to which the data is to be output in CSV format. The specification of `csv-filename` is mandatory in the first IN CSV statement of a BPRECORD run (e.g. `DISPLAY IN CSV 'BPRECORD.CSV' ...`).

For a detailed description of CSV format output, see the manual “Database Operation”, section “Outputting database information in a neutral format”.

**page-selection**

See “Physical selection (page selection)” on page 212

Overflow pages lying outside of the page selection are also output with the primary CALC pages.

**ALL**

Print all realm CALC pages defined by `page-selection`

**ALL RECORDS**

Print the CALC pages (defined by `page-selection`) of all record types defined with LOCATION CALC

**ALL CALC SEARCHKEYS**

Print the CALC pages (defined by `page-selection`) of all CALC SEARCH keys

**ONLY RECORD record-name**

Print all records or specific records (`rsq-selection`) from the CALC pages defined by `page-selection`

*record-name*

Name of a record type defined with LOCATION CALC
ONLY CALC SEARCH KEY *keyref*

Print all CALC index entries or part (*rsq-selection*) of the CALC index entries from the CALC pages (defined by *page-selection*) of the CALC SEARCH key

*keyref*

Key number of a CALC SEARCH key

(see “CALC-SEARCH-KEY INFORMATION” on page 142)

*rsq-selection*

See “Logical selection (RSQ selection)” on page 212.

**Example 1**

```sql
DISPLAY CALC PAGES PAGE 16 ALL CALC SEARCHKEYS
```

```
*********************************** CALC KEY BUCKETS OF REALM < :H32:$XXXXXX.Shipping.Article-RLM > ***********
=========================== CALC SEARCH KEY, KEY REF 14, SET NAME < SYS_ARTICLE > ===========================
-------------------------------------- REALM REF 11 BNR 16 ----- ACT KEY X'0B000010' -------------------------------------
PAGE INFO: TYPE 0 -- FREE SPACE SIZE 10, DISPL 30 -- NR OF PAGE INDICES 0 -- DISPL TO CALC TABLE HEADER 30
CALC KEY TABLE: MAX ENTRIES 220 -- ACT ENTRIES 25 -- OVERFLOW BUCKET NEXT 0, PRIOR 0

-1- ( 1) (00000000) F2F3F0F1 F0F7F3F6 RSQ 1 — PPP 5, 13 23010736
-2- ( 1) (00000000) F2F3F0F1 F0F7F4F2 RSQ 4 — PPP 5, 13 23010742
-3- ( 1) (00000000) F2F3F0F1 F0F7F4F8 RSQ 7 — PPP 5, 13 23010748
-4- ( 1) (00000000) F2F3F2F1 F0F7F3F6 RSQ 8 — PPP 5, 14 23210738
-5- ( 1) (00000000) F2F3F2F1 F0F7F4F0 RSQ 9 — PPP 5, 14 23210740
-6- ( 1) (00000000) F2F3F2F1 F0F7F4F6 RSQ 12 — PPP 5, 14 23210746
-7- ( 1) (00000000) F2F3F3F1 F0F8F3F8 RSQ 21 — PPP 5, 15 23310838
```
Example 2

DISPLAY CALC PAGES ALL PAGES ONLY RECORD ART-DESCR ALL RSQS
(record type defined with LOCATION CALC)

********************************************************************* SCHEMA NAME < MAIL-ORDERS > *********************************************************************
********************************************************************* CALC KEY BUCKETS OF REALM < :H32:$XXXXXXXX.SHIPPING.CLOTHING > *********************************************************************
********************************************************************* CALC RECORD, REC REF 8, RECORD NAME < ART-DESCR > *********************************************************************

REALM REF 5 BNR 2 ACT KEY X'05000002'

PAGE INFO: TYPE 0 -- FREE SPACE SIZE 20, DISPL 40 -- NR OF PAGE INDICES 0 -- DISPL TO CALC TABLE HEADER 40
CALC KEY TABLE: MAX ENTRIES 79 -- ACT ENTRIES 2 -- OVERFLOW BUCKET NEXT 0, PRIOR 0

-1-
( 1) (00000000) C6D3D6E6 C9D5C740 D1C5D9E2 C5E840C4 D9C5E2E2 40404040 40404040 40404040 FLOWING JERSEY DRESS
  ( 33) (00000200) 40404040 40404040
    RSO 8 --- PPP 5, 16

-2-
( 1) (00000000) D7D6D3D6 40C4D9C5 E2E24040 40404040 40404040 40404040 POLO DRESS
  ( 33) (00000200) 40404040 40404040
    RSO 11 --- PPP 5, 19

REALM REF 5 BNR 3 ACT KEY X'05000003'

PAGE INFO: TYPE 0 -- FREE SPACE SIZE 20, DISPL 40 -- NR OF PAGE INDICES 0 -- DISPL TO CALC TABLE HEADER 40
CALC KEY TABLE: MAX ENTRIES 79 -- ACT ENTRIES 4 -- OVERFLOW BUCKET NEXT 0, PRIOR 0

-1-
( 1) (00000000) D1C5D9E2 C5E840C3 D9C5E2C5 40404040 E2E24040 40404040 40404040 40404040 JERSEY CREPE DRESS
  ( 33) (00000200) 40404040 40404040
    RSO 7 --- PPP 5, 15

-2-
( 1) (00000000) D7D3C5C1 E3C5C440 C4D9C5E2 E2404040 E3C840D1 C3D2C5E3 40404040 40404040 PLEATED DRESS WITH JACKET
  ( 33) (00000200) 40404040 40404040
    RSO 6 --- PPP 5, 14

-3-
( 1) (00000000) E2E4D4D4 C5D940C4 D9C5E2E2 40E6C9E3 C840D1C1 C3D2C5E3 40404040 40404040 SUMMER DRESS WITH JACKET
  ( 33) (00000200) 40404040 40404040
    RSO 1 --- PPP 5, 12

-4-
( 1) (00000000) E36D6880 D7C1D9E4 40E2E4D4 D4C5D940 C4D9C5E2 E2404040 40404040 40404040 TWO-PART SUMMER DRESS
  ( 33) (00000200) 40404040 40404040
    RSO 2 --- PPP 5, 13

REALM REF 5 BNR 4 ACT KEY X'05000004'

PAGE INFO: TYPE 0 -- FREE SPACE SIZE 20, DISPL 40 -- NR OF PAGE INDICES 0 -- DISPL TO CALC TABLE HEADER 40
CALC KEY TABLE: MAX ENTRIES 79 -- ACT ENTRIES 2 -- OVERFLOW BUCKET NEXT 0, PRIOR 0

-1-
( 1) (00000000) C6D3D6E6 C9D5C740 D1C5D9E2 C5E840C4 D9C5E2E2 40404040 40404040 40404040 FLOWING JERSEY DRESS WITH
  ( 33) (00000200) 40404040 40404040
    RSO 9 --- PPP 5, 17

-2-

DISPLAY CALC statement

CALC RECORD
Hash area of a record type defined with LOCATION CALC

REC REF
Number of the record type

RECORD NAME
Name of the record type

CALC SEARCH KEY
Hash area of a SEARCH key defined with USING CALC

KEY REF
Number of the key

SET NAME
Name of the set to which this key belongs

PAGE INFO
Page header (see PRINT statement);
in CALC pages, DISPL TO END OF PAGE is replaced by:

DISPL TO CALC TABLE HEADER
Displacement to the CALC key table header

LOGICAL RECORDS
Print out of records with the following options:

PAGE INDEX
Page index entry (see PRINT statement)

SCD
Set connection data (see PRINT statement)

CALC KEY TABLE
BPRECORD prints the following information from the CALC key table header:

MAX ENTRIES
Maximum number of entries possible

ACT ENTRIES
Actual number of current entries
OVERFLOW BUCKET
Linkage to overflow pages:

NEXT n
Page number of the next overflow page
0: No overflow page exists

PRIOR m
Page number of the preceding page
0: Primary page
Print data pages (DISPLAY DATA)

\[
\text{DISPLAY [IN CSV [csv-filename]] DATA PAGES page-selection}
\]

\[
\begin{align*}
\text{ALL} & \{ \{ \text{RECORDS} \} \} \\
\text{TABLES} & \{ \text{RECORD record-name} \} \\
\text{ONLY} & \{ \{ \text{OWNER record-name} \} \} \text{ rsq-selection} \\
\text{TABLES OF} & \{ \{ \text{SET set-name} \} \} \text{ KEY keyref} \\
\end{align*}
\]

IN CSV
BPRECORD also outputs the data in CSV format.

csv-filename
Name of the file to which the data is to be output in CSV format. The specification of csv-filename is mandatory in the first IN CSV statement of a BPRECORD run (e.g. DISPLAY IN CSV 'BPRECORD.CSV' ...).

For a detailed description of CSV format output, see the manual “Database Operation”, section “Outputting database information in a neutral format”.

page-selection
See “Physical selection (page selection)” on page 212.
Overflow pages lying outside of the page selection are also output with the primary table pages.

ALL Print all realm data pages defined by page-selection

ALL RECORDS Print all record types from the data pages defined by page-selection

ALL TABLES Print all tables from the data pages defined by page-selection

ONLY RECORD record-name
From the data pages defined by page-selection, print the records or certain record (rsq-selection) of the record type record-name

record-name
Name of a record type which has not been defined with LOCATION MODE CALC

ONLY TABLES OF
From the data pages defined by page-selection, print out the tables or certain table entries (rsq-selection)
OWNER record-name
Of the owner record type record-name

SET set-name
Of the set set-name

KEY keyref
Of the key with the number keyref (see page 139).

rsq-selection
See “Logical selection (RSQ selection)” on page 212. An RSQ selection is of no use for system sets and is ignored if present.

Example 1

DISPLAY DATA PAGES PAGE 19 ALL RECORDS
(only data records)

For descriptions of PAGE INFO, PAGE INDEX and SCD, see page 217.
**Example 2**

DISPLAY DATA PAGES ALL PAGES ALL TABLES  
(only tables)

****************************************************************************** DATA / TABLE PAGES OF REALM < :H32:XXXXXXXX.SHIPPING.ARTICLE-RLM > ******

-- REALM REF 11 BNR 31 ----- ACT KEY X'OB00001F' --

PAGE INFO: TYPE 0 — FREE SPACE SIZE 0, DISPL 32 — NR OF PAGE INDICES 1 — DISPL TO END OF PAGE 4000

LOGICAL RECORDS:
-1- PAGE INDEX: DB_KEY 0, 5 — COL-NR 1 — TABLE REC — DISPL 32
 TABLE: MAX ENTRIES 112 — TABLE NEXT 0 — TABLE DESCR X'40' — NEXT HIGHER LEVEL 0
 ACT ENTRIES 5 — PRIOR 0 — LEVEL NR 0 — LAST ENTRY 31

-1-  RSO  2 — PPP 8, 4
 ( 1) (00000000) C2C1E5C1 D9C9C1D5 40C2C5C5 D9404040 40404040 40404040 40 BAVARIAN BEER

-2-  RSO  4 — PPP 5, 18
 ( 1) (00000000) C3D6D5E3 C5D4D7D6 D9C1D9E8 40C3D3D6 E3C8C9D5 C7404040 40 CONTEMPORARY CLOTHING

-3-  RSO  1 — PPP 5, 12
 ( 1) (00000000) C5D3C5C7 C1D5E340 C3D3D6E3 C8C9D5C7 40404040 40404040 40 ELEGANT CLOTHING

-4-  RSO  3 — PPP 8, 5
 ( 1) (00000000) D3C5D4D6 D5C1C4C5 40404040 40404040 40404040 40404040 40 LEMONADE

-5-  RSO  5 — PPP 8, 6
 ( 1) (00000000) E8D6C7C8 E4D9E340 40404040 40404040 40404040 40404040 40 YOGHURT

[ Table header ]
[ Associated table entries ]

**TABLE**

From the header of the table, BPRECORD prints the following information:

**MAX ENTRIES**  
Self-explanatory

**ACT ENTRIES**  
Number of current entries

**TABLE**  
Linkage of table pages

**NEXT n**  
Page number of the next table page

**PRIOR m**  
Page number of the preceding table page;

0: No next or prior table page exists

232
TABLE DESCR
Description of the table
$2^7 = 1$: List
$2^6 = 1$: Multi-level table
$2^5 = 1$: Table ATTACHED TO OWNER
$2^4 = 1$: Duplicates table

LEVEL NR
Level of the table

NEXT HIGHER LEVEL
Page number of the page of the next higher level

LAST ENTRY
Page number of the last page of the basic level

Terminate BPRECORD (END)
7.4 Command sequence to start BPRECORD

The command sequence described here is based on the assumption that UDS/SQL was installed with IMON (see the section “START commands of the UDS/SQL programs” in chapter 2 of the “Creation and Restructuring” manual).

01 /ADD-FILE-LINK LINK-NAME=DATABASE,
   FILE-NAME=[:catid:]$userid.dbname.DBDIR[.copy-name]
02 /SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL,VERSION=version,SINGLE=*,SCOPE=*TASK
03 /START-UDS-BPRECORD
04 bprecord-statements
05 END

01 In this case specifying :catid: is permitted (see the “Database Operation” manual).
02 The version of the utility routine is selected. Specification of the version is generally recommended, since several UDS/SQL versions can be installed in parallel.
03 The UDS/SQL utility routine can also be started with the alias BPRECORD.
8 Database reorganization with BREORG

Reorganization is an important part of database maintenance. It helps save not only storage space, but also time.

Storage space can be saved by reducing the size of realms, reducing the size of the Database Key Translation Table (DBTT) of a record type and thereby reducing the maximum permissible number of records, and creating new set tables with a higher occupancy level.

Time can also be saved when accessing the database by reducing the number of overflow pages for hash areas, updating physical pointers which are in the form of Probable Position Pointers (PPP), and creating new set tables with a different occupancy level.

Reorganization may become necessary when a realm has become too small, or more records of a record type are stored than originally planned.

Reorganization should be performed on a regular basis if record types which are specified with LOCATION MODE IS CALC and frequently updated are stored in the database as sort key tables, pointer arrays, chains, or lists. In such cases, updating may render the probable position pointers in the tables incorrect. This would adversely affect access times to the records via the tables (see the “Database Operation“ manual).
8.1 Functions

The BREORG utility routine provides the following functions:

- Define buffer size
- Modify realm size
- Modify record population
- Reorganize CALC areas
- Reorganize tables and set constructs
- Reorganize PPPs

BREORG functions can be applied on the user realms of the database as well as the PRIVACY-AND-IQF database in the DBDIR and the compiler database in the DBCOM.
8.2 System environment

To run BREORG, information about the areas, sets, record types and tables to be reorganized is required. BREORG obtains this information from the schema information area (SIA) of the associated schema.

Figure 13: System environment for the MODIFY-REALM-SIZE and MODIFY-RECORD-POPULATION functions
For its REORGANIZE functions, BREORG also requires subschema information (only for the recreation of multi-level list sets). This information is obtained from the Subschema Information Area (SSIA).
**Work files**

For the REORGANIZE functions, BREORG requires different work files on disk. These files are automatically created and stored on public disk under the appropriate user ID and erased again after normal completion of the run.

**Work files for the REORGANIZE-CALC and REORGANIZE-SET statements**

The statements REORGANIZE-CALC and REORGANIZE-SET require two work files using the standard file link names SCRTCH1 and SORTWK:

**SCRTCH1**
- BREORG requires this file in order to reorganize direct and indirect hash areas or multi-level tables.
- It contains an intermediate version of the CALC entries or table lines to be processed.

**SORTWK**
- Requires the SORT used by BREORG for sorting internal evaluation records (see manual “SORT (BS2000/OSD)”).

If these work files are to be explicitly created, they must be assigned the following attributes:

**Work-file-1**

File link name SCRTCH1

Access method = PAM

Primary allocation, calculated as follows:

The data population for buffering can be calculated using the following formulae

- for the reorganization of indirect hash areas:
  \[(12 + \text{key length}) \times \text{number of entries}\]  Bytes

- for the reorganization of direct hash areas:
  \[8 \times \text{number of entries}\]  Bytes

- for the reorganization of multi-level tables:
  \[12 \times \text{number of entries}\]  Bytes
key length
Length of CALC key

number of entries
Number of CALC index entries or occupied table lines

Work-file-2
SORT needs Work-file-2 if there is not enough virtual memory for pre-sorting. The primary allocation should be based on the data population that is to be sorted while taking account of the safety factor recommended by SORT (see the discussion of work files in the manual “SORT (BS2000/OSD)” ). There should always be an appropriate secondary allocation in case it is necessary to extend the storage space.

File link name SORTWK
Access method = PAM

The data population for sorting can be calculated using the following formulae

– for the reorganization of indirect hash areas:
  \((12 + \text{key length}) \times \text{number of entries}\) Bytes

– for the reorganization of direct hash areas:
  \((\text{record length} + \text{key length} + 7) \times \text{number of entries}\) Bytes

– for the reorganization of multi-level tables:
  \(12 \times \text{number of entries}\) Bytes

key length
Length of CALC key

number of entries
Number of CALC index entries or occupied table lines

record length
Length of records or table lines. Secondary allocation in case the storage space must be enlarged; secondary should be not less than 120, and not zero.
If you do not create the two work files yourself, BREORG sets them up automatically with the following names and sizes:

- **UTI.tsn.SCRTCH1**
  - Size: (360,360)
  - Used for REORGANIZE-SET and for REORGANIZE-CALC
- **UTI.tsn.SORTWK**
  - Size: (120,120)

*tsn* stands for the task sequence number of the current task.

**Work files for the REORGANIZE-POINTERS statement**

With the REORGANIZE-POINTERS statement, BREORG uses work files for the record types and an additional work file for sorting.

You can also create the work files for the record types yourself via the file name, and the work file for sorting via the file link name.

**Work files for the record types**

- **File names UTI.BREORG.dbname.xxx.yyyyy**
  - *dbname* Name of the database
  - *xxx* Realm number of the specified realm
  - *yyyyy* Number of the record type whose PPPs are updated in the realm; *yyyyy*=0 is used for PPPs in system sets if required.

Access method: SAM

The data population for buffering can be calculated using the following formula

\[ \text{number of ppps} \times 11 \text{ Bytes} \]

If you do not set up the files yourself, then BREORG uses the size of the DBTT for the record type *yyyyy* together with the size of the realm *xxx* to calculate the expected data population.

The minimum size is based on 5000 objects for buffering.
• File name UTI.BREORG.dname.xxx.00001
  
  *dbname*  Name of the database  
  *xxx*  Realm number of the specified realm  
  
  Access method: SAM  
  
  The user schema does not contain any record type with record type number 1. All the updated PPPs, sorted by their position in the realm, are stored in the work file with record type number 1. The size of this file therefore depends on the total size of required individual files UTI.BREORG.dname.xxx.yyyy (yyyyy = 0 bzw. yyyyy > 1).  
  
  The work files are deleted once the PPP update has been completed.

**Work file for sorting**

SORT needs this work file if there is not enough virtual memory for pre-sorting. The primary allocation should be based on the data population that is to be sorted while taking account of the safety factor recommended by SORT (see the discussion of work files in the manual “SORT (BS2000/OSD)”). There should always be an appropriate secondary allocation in case it is necessary to extend the storage space.  

File link name: SRT1WK  
Access method: PAM  
Approximate size: maximum size of all files UTI.BREORG.dname.xxx.00001  
If the work file was created explicitly for sorting, it must also be deleted again explicitly if need be.
8.3 Database saving

If you have set up ALOG files and enabled AFIM logging, BREORG records after-images for all functions.

If ALOG files are missing when BREORG starts (even though AFIM logging was enabled), BREORG terminates with an error message before processing begins.

If there is an error involving an ALOG file while BREORG is running, further logging is suppressed, and BREORG terminates after execution of the current statement. This generally results in the creation of a logging gap.

Console message on AFIM logging

BREORG examines task switch 29. If this switch is set, the console message LOGGING STOPPED FOR DATABASE dbname is issued when AFIM logging is aborted.
8.4 BREORG statements

Rules for input of statements

Incorrectly specified statements can be corrected interactively via SDF.

If the input is not made via the SDF mask, it is not always legal to enter any statement at any time, so some statements (e.g. OPEN-DATABASE) may be rejected.

The following statements are executed in the order given by the user, but only after the END statement is specified:

1. MODIFY-REALM-SIZE
2. MODIFY-RECORD-POPULATION,
3. REORGANIZE-CALC
4. REORGANIZE-POINTERS
5. REORGANIZE-SET.

If a statement containing an error is encountered when processing statements in batch mode, all statements until the first incorrect statement will be executed.

<table>
<thead>
<tr>
<th>Statement</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALLOCATE-BUFFERPOOL</td>
<td>Define buffer size (in Mbytes)</td>
</tr>
<tr>
<td>END</td>
<td>Terminate input of statements</td>
</tr>
<tr>
<td>MODIFY-REALM-SIZE</td>
<td>Modify realm size</td>
</tr>
<tr>
<td>MODIFY-RECORD-POPULATION</td>
<td>Modify record population</td>
</tr>
<tr>
<td>OPEN-DATABASE</td>
<td>Open database</td>
</tr>
<tr>
<td>REORGANIZE-CALC</td>
<td>Reorganize CALC areas</td>
</tr>
<tr>
<td>REORGANIZE-POINTERS</td>
<td>Reorganize all PPPs contained in one realm</td>
</tr>
<tr>
<td>REORGANIZE-SET</td>
<td>Reorganize tables and set constructs</td>
</tr>
<tr>
<td>SPECIFY-SCHEMA</td>
<td>Specify schema</td>
</tr>
<tr>
<td>SPECIFY-SUBSCHEMA</td>
<td>Specify subschema</td>
</tr>
<tr>
<td>UNDO</td>
<td>Undo statement</td>
</tr>
</tbody>
</table>

Table 18: BREORG statements

The individual statements of BREORG are described below in alphabetical order.
Define buffer size (ALLOCATE-BUFFERPOOL)

The ALLOCATE-BUFFERPOOL statement defines the size of the used buffer pool in Mbytes.

If default values are not to be used for buffer initialization, this statement must be the first statement specified.
After the initial allocation, the ALLOCATE-BUFFERPOOL statement is no longer offered in the SDF mask.
This statement cannot be canceled with the UNDO statement.

<table>
<thead>
<tr>
<th>ALLOCATE-BUFFERPOOL</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUFFER-SIZE = STD / &lt;integer 1..2000&gt;</td>
</tr>
</tbody>
</table>

**BUFFER-SIZE = STD**
The default size of the buffer pool is defined as 1 or 2 Mbytes, depending on the system involved.

**BUFFER-SIZE = <integer 1..2000>**
The size of the buffer pool must lie within the given limits. The maximum value depends on the system configuration and the version of the operating system.
**Terminate input of statements (END)**

The END statement is used to terminate the input of statements. All entered statements are executed after this statement.

The END statement cannot be canceled with the UNDO statement.

```
END
```

This statement has no operands.
MODIFY-REALM-SIZE statement

Modify realm size (MODIFY-REALM-SIZE)

The MODIFY-REALM-SIZE statement can be used to specify either the absolute size of a realm of the database or a relative change.

<table>
<thead>
<tr>
<th>MODIFY-REALM-SIZE</th>
</tr>
</thead>
<tbody>
<tr>
<td>REALM-NAME = &lt;realm-name&gt;</td>
</tr>
<tr>
<td>REALM-SIZE = &lt;integer 1..16777216&gt; / *RELATIVE(...) / MINIMUM</td>
</tr>
<tr>
<td>*RELATIVE(...)</td>
</tr>
<tr>
<td>DIFFERENCE = &lt;integer -16777216..16777216&gt;</td>
</tr>
</tbody>
</table>

REALM-NAME = <realm-name>
Name of the realm which is to be modified.

REALM-SIZE = <integer 1..16777216>
The new size is equivalent to the specified value in UDS/SQL database pages.

REALM-SIZE = *RELATIVE (...)
The new size is calculated from the old size and the specified difference (which may be a positive or negative value), but cannot be less than the size attained by specifying MINIMUM.

DIFFERENCE = <integer -16777216..16777216>
Difference with respect to the old realm size in UDS/SQL database pages.

REALM-SIZE = MINIMUM
The realm is reduced by the number of empty pages at the end.

Enlarging a realm

The physical enlargement of the file <realm-name> is requested by BREORG from the BS2000 DMS. The new pages are collected by the free place administration of the realm. If there is not enough free space (as determined by the free place administration, FPA) for the realm, then new free place administration tables (FPA-Extents) are created as needed.

When the DBDIR or DBCOM is expanded, the new, empty pages are always formatted. Whether or nor new, empty pages are formatted when expanding a user realm depends on the state of the PPPs:

- If the PPPs were updated in all user realms with the REORGANIZE-POINTERS statement, then the new, empty pages are not formatted.
- If a PPP reference in the new pages cannot be ruled out, then the new, empty pages are formatted.
If you are using private disks for storage, you should ensure that the disk provides sufficient space for enlarging the realms. If the disk on which the realm ends does not have enough space for a realm enlargement, a continuation disk must be assigned for the file before the BREORG run.

Reducing a realm

BREORG reduces the size of the realm in the following manner: First, the achievable realm size is determined from the results of the MODIFY-REALM-SIZE statement and the data in the realm. Unneeded FPA sections are released. FPA sections still needed are generally moved to the beginning of the realm (if this is possible). BREORG then requests the physical reduction of the realm via the DMS of BS2000, so no CREATE-FILE statement is required to reduce the area.

The realm DBDIR may only be modified under the PRIVACY-AND-IQF-SCHEMA. The name DBDIR is internally converted to the realm name DATABASE-DIRECTORY, the name DBCOM to the realm name DATABASE-COMPILER-REALM.

Execution messages

On executing a MODIFY-REALM-SIZE statement, the results for the new free place administration (FPA) of a realm are output as follows:

```
***** RESULTS OF FPA-REORGANISATION OF AREA area name
 NEW FPA FIRST PAGE     : area ref - page no
 NEW FPA LAST PAGE      : area ref - page no
 NEW NR OF EXTENTS      : number extents
 NEW FPA SIZE           : number pages
 NEW NR OF PAGES        : number pages
 NR OF DATABASE ACCESSES: number physical io
```

*area-name*
Name of the enlarged realm or reduced realm

*area ref - page no*
For FPA FIRST PAGE: Smallest act-key of all act-keys of FPA pages (not necessarily identical to the beginning of the FPA)
For FPA LAST PAGE: Largest act-key of all act-keys of FPA pages (not necessarily identical to the end of the FPA)
number extents
   Number of FPA extents

number pages
   For NEW FPA SIZE: Number of pages in the new FPA area
   For NEW NR OF PAGES: New page count for the realm

number physical io
   Number of physical input and output operations
Modify record population (MODIFY-RECORD-POPULATION)

The MODIFY-RECORD-POPULATION statement can be used to change the permissible number of records of a record type.

| MODIFY-RECORD-POPULATION
<table>
<thead>
<tr>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RECORD-NAME = &lt;record-name&gt;</td>
</tr>
<tr>
<td>RECORD-POPULATION = &lt;integer 1..2147483647&gt; / *RELATIVE(...) / MINIMUM</td>
</tr>
</tbody>
</table>
| *
| DIFFERENCE = <integer -2147483647..2147483647> |

**RECORD-NAME = <record-name>**
Name of the record type for which the record population is to be changed.

**RECORD-POPULATION = <integer 1..2147483647>**
The number of permissible DBTT entries for the record type is equivalent to the specified value.

**RECORD-POPULATION = *RELATIVE (...)**
The new size of the DBTT is calculated from the old size and the specified difference (which may be a positive or negative value), but cannot be less than the size attained by specifying MINIMUM.

**DIFFERENCE = <integer -2147483647..2147483647>**
Difference with respect to the old DBTT size; as a number of DBTT entries.

**RECORD-POPULATION = MINIMUM**
The DBTT of the record type is reduced to the smallest possible value.

Entries that are invalid for databases with a 2-kbyte format cannot be detected via SDF if the entered values lie within the ranges specified above.
Since these databases are subject to additional checks, the old limits are still applicable to them.

Since the DBTT always occupies all pages that are reserved for it entirely, it is conceivable that after a BREORG run in which an increase is requested, more records can be stored than were actually specified in the statement. In the reverse case, if you use the statement to deallocate fewer DBTT entries than are actually contained in a DBTT page of the specified record type, then the maximum permissible number of records in this record type stays the same.
The DBTT of a record type always occupies consecutive pages within a realm. For this reason, when the number of records is changed, BREORG searches from the beginning of a realm for an appropriate area of free pages - treating the old DBTT as a free area - and relocates the new DBTT to it.

Since entire pages are always reserved for the DBTT, the number of desired entries is rounded up to full pages. However, if the maximum RSQ is exceeded in the process, the number is rounded down to full DBTT pages.

The SSIA-RECORD may only be modified under the PRIVACY-AND-IQFSCHEMA.

When a database is reorganized, DBTT extents may be created independently of the activation of online DBTT extension. An increase in the size of the DBTT due to BREORG is implemented in DBCOM and DBDIR by enlarging the existing DBTT. In the user realm, the DBTT is enlarged by BREORG by means of DBTT extents if the target DBTT has a total size greater than 128 PAM pages. If the target DBTT is smaller than or equal to 128 PAM pages then BREORG always implements it as a DBTT base, i.e. as a single unit. The corresponding messages inform you of the results of increases or reductions in the size of DBTTs.

Execution messages

On executing a MODIFY-RECORD-POPULATION statement, the results of the DBTT reorganization of the record type are output as follows:

***** BEGIN OF DBTT-SIZE-MODIFICATION AT hh:mm:ss
***** RESULTS OF DBTT-REORGANIZATION OF RECORD record-name
NEW DBTT FIRST PAGE : area ref - page no
NEW DBTT LAST PAGE : area ref - page no
NEW NR OF EXTENTS : number extents
NEW DBTT SIZE : number pages
NEW NR OF DBTT ENTRIES : number entries
***** END OF DBTT-SIZE-MODIFICATION AT hh:mm:ss

record-name
Name of the modified record type

area ref - page no
Act-key of the first or last DBTT page

number extents
New number of DBTTextents

number pages
New number of DBTT pages
number entries
    New number of records

If no modification has taken place, then the value NOT CHANGED is output instead of the numerical specification.
Open database (OPEN-DATABASE)

The OPEN-DATABASE statement defines the database to be processed by subsequent statements of BREORG.

This statement is not offered if the database is assigned using LINK=DATABASE.

<table>
<thead>
<tr>
<th>OPEN-DATABASE</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATABASE-NAME = &lt;dbname&gt;</td>
</tr>
<tr>
<td>SCHEMA-NAME = *STD / &lt;schema-name&gt;</td>
</tr>
<tr>
<td>USER-IDENTIFICATION = *OWN / &lt;userid&gt;</td>
</tr>
</tbody>
</table>

DATABASE-NAME = <dbname>
Name of the database. You can only process a database that is cataloged under your own user ID. A database under a foreign user ID can only be processed from the system administrator ID TSOS.

SCHEMA-NAME = *STD
The name of the user schema that was defined for the database is used.

SCHEMA-NAME = <schema-name>
_schema-name_ specifies the database schema for whose objects the BREORG statements are to be executed.

Possible values:
PRIVACY-AND-IQF-SCHEMA
COMPILER-SCHEMA
Name of the user schema

USER-IDENTIFICATION = *OWN
The database is located under the user’s own user ID.

USER-IDENTIFICATION = <userid>
The specification of a foreign user ID is only permitted under the system administrator ID TSOS.

In order to process a database, BREORG requires information on the realms, record types, and set relations in the database. The schema name allows BREORG to access the SIA in which this information is contained and subsequently modified as required.
Reorganize CALC areas (REORGANIZE-CALC)

The REORGANIZE-CALC statement is used to reorganize CALC areas that belong to a particular record type. These are:
- areas created by means of LOCATION MODE CALC
- SEARCH KEY USING CALC on record type level
- SEARCH KEY USING CALC in singular sets in which the record type is a member

<table>
<thead>
<tr>
<th>REORGANIZE-CALC</th>
</tr>
</thead>
<tbody>
<tr>
<td>RECORD-NAME = &lt;record-name&gt;</td>
</tr>
</tbody>
</table>
| ,CALC-RECORD = NONE / list-poss(6): *WITHIN-POPULATION(…)
  *WITHIN-POPULATION(…)
    REALM = *ALL / <realm-name>
    ,POPULATION = UNCHANGED / <integer 1..2147483647> |
| ,CALC-SEARCHKEY = NONE / list-poss(30): *KEY-POPULATION(…)
  *KEY-POPULATION(…)
    KEY-REF = ALL / <integer 1..65535> |
    ,POPULATION = STD / UNCHANGED / <integer 1..2147483647> |

RECORD-NAME = <record-name>
Name of the record type whose CALC areas are to be reorganized.

CALC-RECORD = NONE
LOCATION MODE CALC areas are not reorganized.

CALC-RECORD = list-poss(6): *WITHIN-POPULATION (…)
The LOCATION MODE CALC areas located in the specified REALM or REALMs are reorganized.

  REALM = *ALL
All CALC areas are reorganized.

  REALM = <realm-name>
Only the CALC area that is located in the specified REALM is reorganized.

  POPULATION = UNCHANGED
Only the probable position pointers (PPP) are updated.
A PPP update is useful for indirect LOCATION MODE CALC areas.
**POPULATION = <integer 1..2147483647>**

The affected CALC area is recreated. The number of specified entries is converted into a number of pages, and the number of pages is then rounded up to the next primary number (= size of the hash area in UDS/SQL pages).

In indirect CALC areas, the PPPs are updated as well.

**CALC-SEARCHKEY = NONE**

The SEARCH KEY USING CALC areas that belong to the record type are not reorganized.

**CALC-SEARCHKEY = list-poss(30): *KEY-POPULATION (...)**

The CALC areas of the specified KEYs are reorganized.

- **KEY-REF = ALL**
  
  All CALC SEARCH KEY areas are reorganized.

- **KEY-REF = <integer 1..65535>**
  
  Only the CALC SEARCH KEY area with the specified KEY REF is reorganized. The KEY REF can be obtained from the BPSIA log.

**POPULATION = STD**

The affected CALC area is recreated. If a LOCATION MODE CALC area is present, the new size of the SEARCH KEY USING CALC area is calculated from the LOCATION MODE CALC area, or the sum of the LOCATION MODE CALC areas if distributed CALC areas are involved. If no LOCATION MODE CALC area exists, the size is based on the DBTT size of the record type.

**POPULATION = UNCHANGED**

Only a PPP update is performed.

**POPULATION = <integer 1..2147483647>**

The affected CALC area is recreated with the specified size. The calculated number of pages is rounded up to the next primary number. In addition, the PPPs are also updated.

If the same REALM or the same KEY REF is entered more than once in a list, the last specification applies.

If *ALL is entered for REALM or KEY REF in a list, the *ALL specification is assumed.
Reorganizing LOCATION MODE CALC areas

Records of a record type which is defined with LOCATION MODE IS CALC are usually stored in the database in a hash area. Their address in this hash area can be calculated by the Database Handler from the respective CALC key and the size of the area.

Only table entries consisting of the CALC key, the RSQ and the PPP are stored in the “indirect” hash area. Such indirect hash areas are generated for a LOCATION MODE CALC specification if the record type is a member in a set specified using MODE IS LIST or if PLACEMENT OPTIMIZATION or COMPRESSION FOR ALL was specified for it in the SSL.

For reorganization, BREORG calculates the size of the new hash area based on the POPULATION specification and reserves the appropriate number of pages. For each record or table entry, it then determines the address in the newly allocated hash area and relocates it there. After reorganization, the pages of the previous hash area are then available for other purposes.

This makes it possible to change the distribution of the entries in the hash area and so avoid the creation of overflow pages. You should print out an overview of the number of primary pages and the occupancy level of the overflow pages in the newly created hash area after the reorganization.

The number of entries in the DBTT of the record type in question is not altered by BREORG. However, in the case of a direct hash area, it does enter the new physical address of the respective record in column 0 of the DBTT which contains the physical addresses of all records.
Reorganizing CALC SEARCH KEY areas

A CALC SEARCH KEY area does not contain the records themselves, but the table entries. Each entry contains the CALC key, the RSQ (record sequence number) and the probable position pointer (PPP, realm number and page number) of the corresponding record.

Three cases must be distinguished when reorganizing CALC SEARCH KEYs:

1. POPULATION = STD has been specified:
   In this case, BREORG determines the new size of the CALC SEARCH KEY area itself. If LOCATION MODE IS CALC has been specified for the record type, this size - or the sum of all areas for a distributed record type is used to determine the population. Otherwise, the size of the DBTT (number of entries) is used as the value for POPULATION.

2. POPULATION = UNCHANGED has been specified:
   BREORG updates the probable position pointers of the table entries. The distribution of the table entries in the primary area and overflow pages remains the same.

   Updating of the probable position pointers may, for example, become necessary if the positions of the records as members of a LIST set have been altered during database processing.

3. A new value has been specified with POPULATION = INTEGER ...:
   BREORG uses the POPULATION specification to calculate the size of the new hash area and reserves an appropriate number of contiguous pages. It then relocates the table entries to the newly-assigned pages and updates their probable position pointers. Since BREORG recreates the table entries in each case, the relocation results in a new distribution over the primary area and overflow pages.

   The CALC pages which were originally reserved are deallocated by BREORG during reorganization.

Since the records themselves are not relocated when CALC SEARCH KEY areas are reorganized, the information in the corresponding DBTT remains unchanged.
Determining the size of a new hash area

The number of CALC pages which BREORG newly allocates as a result of the POPULATION specification can be calculated using the following formulas:

- For an indirect hash area
  - for 2 Kbytes
    \[
    \frac{2018}{\text{calc-key-length} + 7} = \text{entries-per-page}^1
    \]
  - for 4 Kbytes
    \[
    \frac{3970}{\text{calc-key-length} + 10} = \text{entries-per-page}^1
    \]
  - for 8 Kbytes
    \[
    \frac{8066}{\text{calc-key-length} + 10} = \text{entries-per-page}^1
    \]

  and
  \[
  \frac{\text{integer} - 1}{\text{entries-per-page}} + 1 = \text{no-of-pages}^2
  \]

- For an direct hash area
  - for 2 Kbytes
    \[
    \frac{2018}{\text{record-length} + \text{calc-key-length} + 15} = \text{entries-per-page}^1
    \]
  - for 4 Kbytes
    \[
    \frac{3970}{\text{record-length} + \text{calc-key-length} + 22} = \text{entries-per-page}^1
    \]

^1 Round down the result
^2 If the result is not a primary number, it is rounded up to the next higher primary number.
for 8 Kbytes

\[
\frac{8066}{(\text{record-length} + \text{calc-key-length} + 22)} = \text{entries-per-page}^1
\]

and

\[
\frac{\text{integer} - 1}{\text{entries-per-page}} + 1 = \text{no-of-pages}^2
\]

no-of-pages
Number of pages in the hash area

calc-key-length
Length of CALC key (see page 133)

integer
New quantity of data records as per POPULATION specifications

record-length
Length of the record type (user section and system section) (see page 129ff)

entries-per-page
Number of entries (records or CALC table entries) per page

Note that a sufficiently large number of contiguous pages (no less than no-of-pages) must be provided for the new hash area in the realm affected by reorganization. Since the old hash area can be re-used, it may be viewed as a free area.

---

1 Round down the result
2 If the result is not a primary number, it is rounded up to the next higher primary number
Execution messages

On executing a REORGANIZE-CALC statement, the results of the CALC reorganization of the record type and/or of the CALC SEARCH KEYs in the set are output as follows:

```
***** RESULTS OF CALC-REORGANIZATION OF
{ RECORD record-name
  SEARCH-KEY IN SET set-name }

NEW CALC BEGIN : area ref - page no
NEW NR OF PRIMARY BUCKETS : number pages
NEW NR OF OVERFLOW BUCKETS: number pages
NR OF DATABASE ACCESSES : number physical io
```

- `record-name` Name of the reorganized record type
- `set-name` Name of the reorganized set
- `area ref - page no` Act-key of the first CALC page
- `number pages` For NEW NR OF PRIMARY BUCKETS: New number of CALC BUCKETS
  For NEW NR OF OVERFLOW BUCKETS: New number of overflow pages
- `number physical io` Number of physical input and output operations
Reorganize all PPPs in a realm (REORGANIZE-POINTERS)

The REORGANIZE-POINTERS statement can be used to update all the PPS contained in a user realm.

<table>
<thead>
<tr>
<th>REORGANIZE-POINTERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>REALM-NAME = &lt;realname&gt;</td>
</tr>
</tbody>
</table>

REALM-NAME = <realname>
Name of the realm whose PPPs are to be updated. The statement can be specified several times, one after the other, for different realms in the same BREORG run.

REORGANIZE-POINTERS is significantly faster than reorganizing the PPPs using the other REORGANIZE functions. If all the realms in the database are handled in this way, there is no need to format the new pages when subsequently extending the realms with MODIFY-REALMSIZE or online via the DBH. This considerably improves performance. There is no performance gain when reducing the size of a realm.

With the REORGANIZE-POINTERS statement, BREORG uses work files for the record types and an additional work file for sorting (see “Work files for the REORGANIZE-POINTERS statement” on page 241).
Reorganize tables and set constructs (REORGANIZE-SET)

The REORGANIZE-SET statement can be used to reorganize set constructs (LIST, CHAIN, POINTER ARRAY) and any SEARCH KEY USING INDEX on set level or record type level. The reorganization of a SEARCH KEY USING INDEX on record type level occurs via the associated implicit set.

You can reorganize
- all set occurrences
- individual set occurrences which you preset via the RSQ of the owner
- areas of set occurrences which you define via an area specification of the owner RSQ

You can reorganize the following tables with the REORGANIZE-SET function to set levels:
- non-indexed and indexed pointer arrays
- non-indexed and indexed lists
- indexed sort key tables
- chains
- indexed SEARCH key tables (also duplicates tables)

You can reorganize indexed SEARCH key tables (also duplicates tables) with the REORGANIZE-SET function to record levels. The name of the implicit set must be specified.

BREORG updates all address references in the specified tables that were stored as probable position pointers. It can also optionally recreate the tables with a different occupancy level. BREORG also updates all probable position pointers in chains.

If you want to reorganize several tables in a BREORG session, you should first carry out the functions that relocate the data records. You should then update the probable position pointers which still refer to the old position of the data records.
REORGANIZE-SET

SET-NAME = <set-name>

,OWNER-SELECTION = ALL / list-poss(30): <integer 1..2147483647> / *RANGE(…)

*RANGE(…)

  FROM-RSQ = <integer 1..2147483647>
  ,TO-RSQ = <integer 1..2147483647>

,KEY-SELECTION = ALL / list-poss(30): <integer 1..32767>

,FILLING = UNCHANGED / <integer 1..100>

SET-NAME = <set-name>
Name of the set or implicit set to be reorganized. The name of the implicit set is constructed
by combining SYS_ and record-name.

The underscore (_) must be entered as a @ sign!

OWNER-SELECTION = ALL
All set occurrences are reorganized.

OWNER-SELECTION = list-poss(30): <integer 1..2147483647> / *RANGE(…)
Owner and set occurrences are reorganized.

<integer 1..2147483647>
The owners with the specified RSQs are reorganized.

*RANGE (…) 
All set occurrences whose owner RSQs lie within the specified ranges are reorganized.

FROM-RSQ = <integer 1..2147483647>
RSQ of the first owner whose SET and/or TABLE OCCURRENCE/S is/are to be
reorganized.

TO-RSQ = <integer 1..2147483647>
RSQ of the last owner whose SET and/or TABLE OCCURRENCE/S is/are to be
reorganized.
**KEY-SELECTION = ALL**
Every SEARCH KEY USING INDEX and the set construct are reorganized.

**KEY-SELECTION = list-poss(30): <integer 0..32767>**
Every SEARCH KEY USING INDEX for which a KEY REF is specified is reorganized. The KEY REF can be determined from the BPSIA log.

If the set constructs CHAIN, LIST and POINTER ARRAY were not defined with a SORTED INDEXED BY specification, no KEY REF will have been entered in the BPSIA log. If this is the case, the value 0 must be specified for the KEY REF (only permitted in this situation). The set is then reorganized.

**FILLING = UNCHANGED**
Only the PPPs are updated in the tables or set constructs.

**FILLING = <integer 1..100>**
The tables are reorganized with the specified filling ratio.

The REORGANIZE-SET function reorganizes tables (ASC/DESC KEY, SEARCH KEY) and chains. In this case, reorganization means that BREORG updates the PPPs (probable position pointers)

- in the sort key table entries and the SEARCH key table entries,
- in the SCD (set connection data) of records in chains (forwards/backwards chaining), and
- in the SCD of data pages with owner links (PHYSICALLY LINKED TO OWNER)
- in the SCD of owner records with table links (WITH PHYSICAL LINK)

or sets up new tables.

The linkage of a table to the owner in ... WITH PHYSICAL LINK ... is an Act-key.

If FILLING = integer... has been specified, BREORG reorganizes all tables and fills the new table pages with the available updated entries up to the specified percentage. In multi-level tables, the specified percentage applies only to the main level (level 0). On level 1, 95% of the table is filled; on every other level, one table entry is left free. In addition, BREORG updates the entries in the DBTT of the owner record type (column number >0).

In the case of an ASC or DESC key table of a set with MODE IS LIST, the records themselves are in the table pages, i.e. the records are relocated when such a table is created. In this case BREORG also updates the DBTT entries in column 0 of the DBTT of the record type concerned.
The following overview shows which probable position pointers (PPP) and tables can be reorganized using the REORGANIZE SET function (see the “Design and Definition” manual).

<table>
<thead>
<tr>
<th>DDL and SSL statements</th>
<th>Probable Position Pointer (PPP)</th>
<th>Table</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Explanation</strong></td>
<td><strong>Updating possible</strong></td>
<td><strong>Type</strong></td>
</tr>
<tr>
<td><strong>MODE IS CHAIN</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ORDER IS FIRST/NEXT/PRIOR SORTED</td>
<td>Owner record includes PP of 1st member record in chain</td>
<td>yes</td>
</tr>
<tr>
<td></td>
<td>Forward chaining of member records with RSQ and PPP</td>
<td>yes</td>
</tr>
<tr>
<td>ORDER IS LAST or LINKED TO PRIOR</td>
<td>Owner record includes PPP of last member record in chain</td>
<td>yes</td>
</tr>
<tr>
<td>LINKED TO PRIOR</td>
<td>Backward chaining of member records with RSQ and PPP</td>
<td>yes</td>
</tr>
<tr>
<td>ORDER IS SORTED INDEXED BY DEFINED KEYS...</td>
<td>Every table entry includes PPP of member record</td>
<td>yes</td>
</tr>
<tr>
<td>ASC/DESC KEY IS...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ORDER IS SORTED INDEXED BY DATABASE-KEY</td>
<td>Every table entry includes PPP of member record</td>
<td>yes</td>
</tr>
<tr>
<td><strong>MODE IS POINTER-ARRAY</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ORDER IS FIRST/LAST/NEXT/PRIOR</td>
<td>Every table entry includes PPP of member record</td>
<td>yes</td>
</tr>
<tr>
<td>ORDER IS SORTED INDEXED BY DEFINED KEYS...</td>
<td>Every table entry includes PPP of member record</td>
<td>yes</td>
</tr>
<tr>
<td>ASC/DESC KEY IS...</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ORDER IS SORTED INDEXED BY DATABASE-KEY or ORDER IS IMMATERIAL</td>
<td>Every table entry includes PPP of member record</td>
<td>yes</td>
</tr>
</tbody>
</table>

Table 19: Overview of options in the REORGANIZE SET function (part 1 of 2)
If an entire table is contained in one page, BREORG does not set up a new table, even if FILLING... has been specified.

When setting up a new table with multiple table pages on level 0, BREORG inserts at least two entries in each table page.

When a database is reorganized, DBTT extents may be created independently of the activation of online DBTT extension. An increase in the size of the DBTT due to BREORG is implemented in DBCOM and DBDIR by enlarging the existing DBTT. In the user realm, the DBTT is enlarged by BREORG by means of DBTT extents if the target DBTT has a total size greater than 128 PAM pages. If the target DBTT is smaller than or equal to 128 PAM pages then BREORG always implements it as a DBTT base, i.e. as a single unit. The corresponding messages inform you of the results of increases or reductions in the size of DBTTs.

### Table 19: Overview of options in the REORGANIZE SET function (part 2 of 2)

<table>
<thead>
<tr>
<th>DDL and SSL statements</th>
<th>Probable Position Pointer (PPP)</th>
<th>Table</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>MODE IS LIST</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ORDER IS FIRST/LAST/</td>
<td>PPP not included</td>
<td></td>
</tr>
<tr>
<td>NEXT/PRIOR</td>
<td>-</td>
<td>single-level list</td>
</tr>
<tr>
<td>ORDER IS SORTED INDEXED</td>
<td>PPP not included</td>
<td></td>
</tr>
<tr>
<td>(DB key or ASC/DESC key)</td>
<td>-</td>
<td>multi-level list</td>
</tr>
<tr>
<td><strong>SEARCH KEY..USING INDEX</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TYPE IS REPEATED-KEY</td>
<td>Every table entry includes PPP of member record</td>
<td></td>
</tr>
<tr>
<td></td>
<td>yes</td>
<td>multi-level SEARCH key table</td>
</tr>
<tr>
<td>TYPE IS DATABASE-KEY-LIST</td>
<td>PPP not included</td>
<td></td>
</tr>
<tr>
<td></td>
<td>-</td>
<td>duplicates table</td>
</tr>
<tr>
<td><strong>MEMBER IS PHYSICALLY LINKED TO OWNER</strong></td>
<td>Member record includes pointer to owner record (PPP)</td>
<td>yes</td>
</tr>
</tbody>
</table>

1 These PPPs are standard with MODE IS CHAIN.
Execution messages

On executing a REORGANIZE-SET statement, the results of the reorganization of a set or a table are output as follows:

***** RESULTS OF SET-REORGANISATION OF SET set-name
NR OF PROCESSED TABLES   : number table occurrences
NR OF PPP UPDATES         : number of actualized ppps
NR OF DATABASE ACCESSSES  : number physical io

set-name
Name of the set

number table occurrences
Number of tables processed in the set occurrences

number of actualized ppps
Number of updated PPPs

number physical io
Number of physical input and output operations
Specify schema (SPECIFY-SCHEMA)

The SPECIFY-SCHEMA statement defines the schema that contains the objects for which the database is to be processed.

This statement is only offered if the database is assigned with LINK=DATABASE; otherwise, the schema name is specified in the OPEN-DATABASE statement.

If the SPECIFY-SCHEMA statement is not entered as the first statement after ALLOCATE-BUFFERPOOL, the user schema is assumed.

After the initial schema specification, the SPECIFY-SCHEMA statement is no longer offered in the SDF mask.

```
SPECIFY-SCHEMA

SCHEMA-NAME = *STD / <schema-name>
```

**SCHEMA-NAME = *STD**

The name of the user schema that was defined for the database is used (default value).

**SCHEMA-NAME = <schema-name>**

`schema-name` specifies the database schema for whose objects the BREORG statements are to be executed.

Possible values:

- PRIVACY-AND-IQF-SCHEMA
- COMPILER-SCHEMA
- Name of the user schema

In order to process a database, BREORG requires information on the size of the realms, the record types, and the set relations in the database. The schema name allows BREORG to access the SIA in which this information is contained and subsequently modified as required.
Specify subschema (SPECIFY-SUBSCHEMA)

The SPECIFY-SUBSCHEMA statement is used to define the subschema that is required for the creation of new multi-level list sets with user-defined keys (i.e. SORTED INDEXED BY DEFINED KEYS).

<table>
<thead>
<tr>
<th>SPECIFY-SUBSCHEMA</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBSCHEMA-NAME = &lt;subschema-name&gt;</td>
</tr>
</tbody>
</table>

**SUBSCHEMA-NAME = <subschema-name>**
Name of a subschema that is included in the database and contains the key description of the LIST set.

If BREORG’s REORGANIZE-SET function is to be used in reorganizing a multi-level list, the description and key of the member record type which is to be reorganized must be obtained from the associated SSIA. BREORG accesses the SSIA on the basis of the subschema name specified in the SPECIFY-SUBSCHEMA statement.

![Info icon] This statement may be specified more than once. It remains in effect for all following statements until the next correctly entered SPECIFY-SUBSCHEMA statement is encountered.
Undo statement (UNDO)

The UNDO statement cancels the last correctly entered statement. Exceptions: ALLOCATE-BUFFERPOOL, END, and UNDO statements.

Each subsequent UNDO statement cancels the preceding statement in the chain.

The UNDO statement itself cannot be reversed with another UNDO statement.

```
UNDO
```

This statement has no operands.
8.5 Command sequence to start BREORG

The command sequence described here is based on the assumption that UDS/SQL was installed with IMON (see the section “START commands of the UDS/SQL programs” in chapter 2 of the “Creation and Restructuring” manual).

01 //ADD-FILE-LINK LINK-NAME=DATABASE,
   FILE-NAME=[:catid:][$userid.]dbname.DBDIR]

02 //CREATE-FILE FILE-NAME=work-file-1[,SUPPORT=*PUBLIC-DISK
   (SPACE=*RELATIVE(PRIMARY-ALLOCATION=primary,
   SECONDARY-ALLOCATION=secondary))/
   ,SUPPORT=*PRIVATE-DISK(VOLUME=archive-no,
   DEVICE-TYPE=device[,SPACE=...])]
   /ADD-FILE-LINK LINK-NAME=SCRTCH1,FILE-NAME=work-file-1,
   ACCESS-METHOD=*UPAM]

03 //CREATE-FILE FILE-NAME=work-file-2[,SUPPORT=*PUBLIC-DISK
   (SPACE=*RELATIVE(PRIMARY-ALLOCATION=primary,
   SECONDARY-ALLOCATION=secondary))/
   ,SUPPORT=*PRIVATE-DISK(VOLUME=archive-no,
   DEVICE-TYPE=device[,SPACE=...])]
   /ADD-FILE-LINK LINK-NAME=SORTWK,FILE-NAME=work-file-2,
   ACCESS-METHOD=*UPAM]

04 /SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL,VERSION=version.SCOPE=*TASK

05 /START-UDS-BREORG

06 //ALLOCATE-BUFFERPOOL BUFFER-SIZE = ...

07 //OPEN-DATABASE DATABASE-NAME = ...

08 //SPECIFY-SCHEMA SCHEMA-NAME = ...

09 //SPECIFY-SUBSCHEMA SUBSCHEMA-NAME = ...

10 ... Further statements of BREORG

11 //END

01, 07 You must specify one of the two statements.

02, 03 These CREATE-FILE commands can be used to create work files for the REORGANIZE-SET or REORGANIZE-CALC function (see page 239). By analogy, you can optionally create the work files for the REORGANIZE-POINTER statement (see page 241).

04 The version of the utility routine is selected. Specification of the version is generally recommended, since several UDS/SQL versions can be installed in parallel.
05  BREORG can be called from any user ID. The UDS/SQL utility routine can also be started with the alias BREORG or START-UDS-REORGANIZATION.

08  The SPECIFY-SCHEMA statement is only offered if a command /ADD-FILE-LINK..., LINK-NAME=DATABASE was issued earlier.

09  Only required for the REORGANIZE-SET function when creating new multi-level list sets with user-defined keys (i.e. SORTED INDEXED BY DEFINED KEYS).
8.6 Examples

Example 1

The realm CLOTHING in the SHIPPING database is to be reduced by 12 PAM pages.

/SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL, VERSION=02.4A00
/START-UDS-BREORG
***** START BREORG (UDS/SQL V2.4 0400 )  2004-08-12 15:41:15
//OPEN-DATABASE DATABASE-NAME=SHIPPING,SCHEMA-NAME=MAIL-ORDERS
//MODIFY-REALM-SIZE REALM-NAME=CLOTHING, REALM-SIZE=*RELATIVE(DIFFERENCE=-12)
//END
***** BEGIN OF REALM-SIZE-MODIFICATION AT 15:41:15
***** RESULTS OF FPA-REORGANIZATION OF AREA CLOTHING
    NEW FPA FIRST PAGE : NOT CHANGED
    NEW FPA LAST PAGE  : NOT CHANGED
    NEW FPA SIZE       : NOT CHANGED
    NEW NR OF PAGES    : 42
***** END OF REALM-SIZE-MODIFICATION AT 15:41:16

***** DIAGNOSTIC SUMMARY OF BREORG

    NO WARNINGS
    NO ERRORS
    NO SYSTEM-ERRORS

***** END OF DIAGNOSTIC SUMMARY
***** NR OF DATABASE ACCESSES : 149
***** NORMAL END BREORG (UDS/SQL V2.4 0400 )  2004-08-12 15:41:16
**Example 2**

The realm ARTICLE-RLM is increased so much that the FPA base is not large enough any more. Exactly 1 FPA extent will be the result.

/SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL, VERSION=02.4A00
/START-UDS-BREORG

****** START BREORG (UDS/SQL V2.4 0400 ) 2004-08-12 15:41:16

//OPEN-DATABASE DATABASE-NAME=SHIPPING,SCHEMA-NAME=MAIL-ORDERS
//MODIFY-REALM-SIZE REALM-NAME=ARTICLE-RLM, REALM-SIZE=*RELATIVE(DIFFERENCE=2000)
//END

****** BEGIN OF REALM-SIZE-MODIFICATION AT 15:41:16

****** RESULTS OF FPA-REORGANIZATION OF AREA ARTICLE-RLM

<table>
<thead>
<tr>
<th>NEW FPA FIRST PAGE</th>
<th>NOT CHANGED</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEW FPA LAST PAGE</td>
<td>11-</td>
</tr>
<tr>
<td>NEW NR OF EXTENTS</td>
<td>1</td>
</tr>
<tr>
<td>NEW FPA SIZE</td>
<td>33</td>
</tr>
<tr>
<td>NEW NR OF PAGES</td>
<td>2064</td>
</tr>
</tbody>
</table>

****** END OF REALM-SIZE-MODIFICATION AT 15:41:16

****** DIAGNOSTIC SUMMARY OF BREORG

NO WARNINGS
NO ERRORS
NO SYSTEM-ERRORS

****** END OF DIAGNOSTIC SUMMARY

****** NR OF DATABASE ACCESSES : 108

****** NORMAL END BREORG (UDS/SQL V2.4 0400 ) 2004-08-12 15:41:16
Example 3

The CALC areas of the ARTICLE record type in the SHIPPING database are to be reorganized. The SEARCH KEY USING CALC areas that belong to the ARTICLE record type are not reorganized.

/SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL, VERSION=02.4A00
/START-UDS-BREORG
***** START BREORG (UDS/SQl V2.4 0400 ) 2004-08-12 15:41:16
//OPEN-DATABASE DATABASE-NAME=SHIPPING,SCHEMA-NAME=MAIL-ORDERS
//REORGANIZE-CALC RECORD-NAME=ARTICLE, CALC-RECORD=*WITHIN-POPULATION(POPULATION=200), -
// CALC-SEARCHKEY=NONE
//END
***** BEGIN OF CALC-REORGANIZATION AT 15:41:17
***** RESULTS OF CALC-REORGANIZATION OF RECORD ARTICLE
NEW CALC BEGIN : 5- 5
NEW NR OF PRIMARY BUCKETS : 2
NEW NR OF OVERFLOW BUCKETS: 0
***** RESULTS OF CALC-REORGANIZATION OF RECORD ARTICLE
NEW CALC BEGIN : 6- 7
NEW NR OF PRIMARY BUCKETS : 2
NEW NR OF OVERFLOW BUCKETS: 0
***** RESULTS OF CALC-REORGANIZATION OF RECORD ARTICLE
NEW CALC BEGIN : 7- 5
NEW NR OF PRIMARY BUCKETS : 2
NEW NR OF OVERFLOW BUCKETS: 0
***** RESULTS OF CALC-REORGANIZATION OF RECORD ARTICLE
NEW CALC BEGIN : 8- 9
NEW NR OF PRIMARY BUCKETS : 2
NEW NR OF OVERFLOW BUCKETS: 0
***** RESULTS OF CALC-REORGANIZATION OF RECORD ARTICLE
NEW CALC BEGIN : 9- 5
NEW NR OF PRIMARY BUCKETS : 2
NEW NR OF OVERFLOW BUCKETS: 0
***** RESULTS OF CALC-REORGANIZATION OF RECORD ARTICLE
NEW CALC BEGIN : 10- 4
NEW NR OF PRIMARY BUCKETS : 2
NEW NR OF OVERFLOW BUCKETS: 0
***** END OF CALC-REORGANIZATION AT 15:41:17
***** DIAGNOSTIC SUMMARY OF BREORG

NO WARNINGS
NO ERRORS
NO SYSTEM-ERRORS
***** END OF DIAGNOSTIC SUMMARY
***** NR OF DATABASE ACCESES : 149
***** NORMAL END BREORG (UDS/SQIL V2.4 0400 ) 2004-08-12 15:41:17
9 Controlling the reuse of database keys and the free place search with BMODTT

The BMODTT utility routine is used to control the free place search and the reuse of database keys.

Reusing database keys

The database keys of deleted records are reusable (REUSE) unless otherwise specified. It is possible to lock deallocated database keys (KEEP) until they are specifically made reusable (REUSE) or all currently locked database keys are made reusable once only (REMOVE). The previous statement executed (REUSE or KEEP) is taken into account here.

The REUSE and KEEP statements act as markers in the schema information area (SIA) of a database, indicating whether the database keys of deleted records are reusable or locked. The 'NO REUSE' column in the BPSIA log provides you with information on the reusability of database keys.

The REMOVE statement deletes the entries for all deleted records of the specified record type from the DBTT even if that type has been marked KEEP.

Unless otherwise specified (i.e. before BMODTT is used for the first time), the database keys of deleted records are reusable. If a record type is deleted, the first database key to be allocated is set to the database key before the first one that is deallocated or to 1 (if the first database key is free).

During restructuring, the default value REUSE is assigned to each record type via BGSIA. Following a BALTER run, locks for deallocated database keys must be set again with BMODTT.

Search for free space

The SET and RESET statements can be used to control how the search for free space is performed, i.e. if the primary task is to optimally utilize the space (SET) or if the 'ATTACHED' and 'PLACEMENT OPTIMIZATION' location specifications set via SSL are to be fulfilled (RESET, default setting).
9.1 System environment

The database administrator is not allowed to invoke BMODTT in the course of an updating session.

Figure 15: BMODTT system environment
9.2 BMODTT statements

<table>
<thead>
<tr>
<th>Statement</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEEP</td>
<td>Lock deallocated database keys</td>
</tr>
<tr>
<td>REMOVE</td>
<td>The locked database keys are released for one-time reuse</td>
</tr>
<tr>
<td>RESET</td>
<td>Search for free space from the end of the realm to the beginning</td>
</tr>
<tr>
<td>REUSE</td>
<td>Deallocate database keys for reuse</td>
</tr>
<tr>
<td>SET</td>
<td>Search for free space from the beginning of the realm</td>
</tr>
</tbody>
</table>

Table 20: BMODTT statements

These statements are explained below in combined formats that reflect their structure.

\[
\{ \text{KEEP} \} \quad \{ \text{REMOVE} \} \quad \{ \text{DBKEY OF RECORD} \} \quad \{ \text{REUSE} \} \quad \{ \text{OF RECORD} \} \quad \{ \text{rec-name-1[,]rec-name-2][,...] \} \quad \{ \text{*ALL [ EXCEPT rec-name-1[,]rec-name-2][,...]} \}
\]

**KEEP**  Deallocated database keys are locked and cannot be reused.

**REMOVE**

The locked database keys are released for one-time reuse with the option of their reusability retained. The highest database key used for the specified records is determined again and stored in the appropriate SIA.

The first database key to be allocated for the affected record types is set to the database key before the first one that is deallocated or set to 1 (if the first database key is free).

**REUSE**

Deallocated database keys can always be reused. This is the default setting before using BMODTT.

\[ \text{rec-name-1[,]rec-name-2][,...} \]

List of affected record types.

*ALL  Affects all record types.

*ALL EXCEPT rec-name-1[,]rec-name-2][,...]  Affects all record types in the database, except for those listed after EXCEPT.
SET
REUSE-FREE-SPACE OF REALM \{realm-1[,realm-2]...\}
RESET

In the search for free space, the search begins with the first page of the realm.

In the search for free space, the search starts with the first page that is followed by only free pages (i.e. a contiguous free area at the end of the realm). This is the default setting before using the BMODTT utility routine.

realm-1[,realm-2]...
List of affected realms.

*ALL  Affects all realms of the database.

*ALL EXCEPT realm-1[,realm-2]...
Affects all realms of the database, except for those listed after EXCEPT.
9.3 Command sequence to start BMODTT

The command sequence described here is based on the assumption that UDS/SQL was installed with IMON (see the section “START commands of the UDS/SQL programs” in chapter 2 of the “Creation and Restructuring” manual).

01 /ADD-FILE-LINK LINK-NAME=DATABASE,
   FILE-NAME=[:catid:][$userid.]dbname.DBDIR[.copyname]

02 /SELECT-PRODUCT-VERSION PRODUCT-NAME=UDS-SQL,VERSION=version.SCOPE=*TASK

03 /START-UDS-BMODTT

04 bmodtt-statements

05 END

01 In this case specifying :catid: is permitted (see the “Database Operation” manual).

02 The version of the utility routine is selected. Specification of the version is generally recommended, since several UDS/SQL versions can be installed in parallel.

03 BMODTT may only be invoked under the database administrator’s user ID. The UDS/SQL utility routine can also be started with the alias BMODTT.

04 The default value (REUSE) is reset whenever BCHANGE or BALTER is used for restructuring.

The BMODTT utility routine is restartable.
Command sequence

BMODTT
Glossary

This Glossary contains the definitions of some of the important terms and concepts used in the UDS/SQL manuals. Terms that appear in *italics* within a particular definition have also been defined in this Glossary.

In cases where two or more terms are used synonymously, a “See” reference points to the more commonly used term in these manuals.

access, contending
See *contending access*.

access, direct
See *direct access*.

access, sequential
See *sequential access*.

access authorization
The rights of a specified user group with regard to access to the *database*. Access rights are defined during live database operation using ONLINE-PRIVACY utility routine or, in offline mode, using the BPRIVACY utility routine.

access path
Means of finding a certain subset of all *records* qualified by a search query, without having to carry out a sequential search of the whole *database*.

access rights
Right of access to a *database* as defined in the BPRIVACY utility routine.

access type
Type of access, e.g. read, update etc.

act-key
(actual key) Actual address of a *page*, consisting of *realm number* and *page number*.
act-key-0 page
First page of a realm; contains general information on the realm such as
– when the realm was created,
– when the realm was last updated,
– internal version number of the realm,
– system break information
– if applicable, warm start information.

act-key-N page
Last ID page of a realm, with the highest page number.
Copy of the act-key-0 page.

address, physical
See act-key or PPP.

administrator task
Task of the independent DBH; The database administrator can control execution of the independent DBH via this task.

AFIM
See after-image.

after-image
Modified portion of a page after its content has been updated.
The DBH writes after-images to the RLOG file as well as the ALOG file.

after-image, ALOG file
The after-images are written to the ALOG file when the ALOG buffer is full. The purpose of the after-images in the ALOG file is to secure the data contained in the database and thus they must be maintained for a long period of time. They are used to reconstruct an original database or update a shadow database.

after-image, RLOG file
After-images are logged in the RLOG file before the updates are applied to the database. The after-images held in the RLOG file are required for warm start only. They are thus periodically overwitten.

 ALOG file
File for securing the data contained in the database in the long term; see after-image.

 ALOG sequence number
See sequence number.
anchor record

*Record* automatically created by UDS/SQL as *owner record* for *SYSTEM sets*. It cannot contain any *items* defined with the *schema DDL* and cannot be accessed.

application

Realization of a job in one or several *user programs* working with UDS/SQL *databases*.

application program (AP)

E.g. *COBOL DML* program or IQS.

area

See *realm*.

ascending key (ASC key)

*Primary key* of a *set*. Defines the sequence of *member records* in the *set occurrences* by ascending key values.

authorization

Identification used for user groups.

authorized users

Specified user groups who are authorized to access the *database*.

backup database

See *shadow database*.

base interface block (BIB)

(Base Interface Block) Standard interface between UDS/SQL and each individual user; it contains, among other things, the *RECORD AREA* (user records as defined in the *subschema*).

before-image

Copy of a *page* taken before its contents are updated. The *DBH* writes before-images to the *RLOG files* during database operation before the updates are applied to the *database*. A prerequisite is that the RLOG files exist.

BFIM

See *before-image*.

BIB

See *base interface block*. 
buffer pool

See system buffer pools and exclusive buffer pool.

CALC key

Key whose value is converted into a relative page number by means of a hash routine.

CALC page

Page of a hash area.

CALC SEARCH key

Secondary key. Used as access path for direct access via hash routine.

CALC table

Table in the direct/indirect CALC page whose entries point to the stored records. Each line contains:
- the CALC key,
- the record sequence number
- the displacement to the related page index entry (direct CALC page) or the probable position pointer (indirect CALC page).

CALL DML

DML that is called by various programming languages (Assembler, COBOL, FORTRAN, PASCAL, PL/1) via the CALL interface.

catalog identifier

Name of the public volume set (PVS) under which the BS2000/UDS/SQL files are stored. The catalog identifier is part of the database or file name and must be enclosed in colons: ":catid:".

chain

Storage mode for a set occurrence in which every record contains a pointer to the subsequent record.

Character Separated Values (CSV)

Output format in which the values are separated by a predefined character.

checkpoint

Consistency point, at which the ALOG file was changed and to which it is possible to return at any time using BMEND utility routine.

check records

Elements which provide information for checking the database. They vary in length from 20 to 271 bytes.
CHECK-TABLE
Check table produced by the DDL compiler during Subschema DDL compilation, and used by the COBOL85 compiler and CALL DML to check whether the DML statements specified in the application program are permitted. It is part of the COSSD or SSITAB module.

COBOL DML
DML integrated in the COBOL85 language.

COBOL runtime system
Runtime system; sharable routines selected by the COBOL compiler for the execution of complex statements.

COBOL Subschema Directory (COSSD)
Provides the COBOL compiler with subschema information for compilation of the DB application programs.

common memory
Shareable memory area used by several different tasks. In UDS/SQL, it always consists of the common pool and the communication pool and, depending on the application, the SSITAB pool (see SSITAB module) if CALL DML is used. If UDS-D is used, it also consists of the distribution pool and the transfer pool.

common pool
Communication area of the independent DBH. Enables DBH modules to communicate with each other. Contains, among other things, an input/output buffer for pages (buffer pools).

communication partners
Tasks or data display terminals.

communication pool
Communication area of the independent DBH for application programs. One of its functions is to store base interface blocks (BIB).

compatible database interface (KDBS)
see KDBS

compiler database
The realms and files of the database which are required by the UDS/SQL compiler. They are
- DBDIR (Database Directory)
- DBCOM (Database Compiler Realm)
- COSSD (COBOL Subschema Directory).
COMPILER-SCHEMA
UDS/SQL-internal schema of the compiler database.

COMPILER-SUBSCHEMA
UDS/SQL-internal subschema of the compiler database.

compound key
Key consisting of several key items.

compression
Only the filled items of a record are stored (see SSL clause COMPRESSION).

configuration
See DB configuration.

configuration identification
Identification in which the database administrator starts the DBH.

configuration name
Freely selectable name of the database configuration for a particular session. The DBH uses it to form:
– the name of the Session Log File,
– the names of the DB status file and its backup copy,
– the names of the RLOG files,
– the names of the temporary realms,
– the names of session job variables,
– the event names of P1 eventing,
– the DCAM application name for the administration,
– the names of the common pools
– the names of the dump files.

connection module
Module that must be linked into every UDS/SQL application program and which establishes the connection with the DBH.

consistency
State of the database without conflicts in the data stored in it.

consistency, logical
State of the database in which the stored data has no internal conflicts and reflects the real-world situation.
**consistency, physical**
State of the database in which the stored data is consistent with regard to correct physical storage, *access paths* and description information.

**consistency, storage**
See *physical consistency*.

**consistency point**
Point (in time) at which the *database* is consistent.

**consistency record**
Administration record with consistency time and date stamps in the *DBDIR*. For an update in a *realm* the *DBH* enters the date and time in the consistency record and in the updated realm. When realms or *databases* are attached for a *session*, the *DBH* uses this time stamp to check the consistency of the realms within each database.

**contending access**
Different *transactions* attempting to access a *page* simultaneously.

**conversation**
*SQL*-specific administration data is retained across transaction boundaries in an *SQL* application. This kind of data administration unit is called a conversation.

**copy**
See *database copy*.

**COSSD**
See *COBOL Subschema Directory*.

**CRA**
(Current Record of Area) *Record* which is marked in the *currency table* as the current record of a particular *realm* (area).

**CRR**
(Current Record of Record) *Record* which is marked in the *currency table* as the current record of a particular *record type* (Record).

**CRS**
(Current Record of Set) *Record* which is marked in the *currency table* as the current record of a particular *set*. 
CRU
(Current Record of Rununit) Record which is marked in the currency table as the current record of the processing chain.

CSV
see Character Separated Values

currency table
The currency table contains:
– CURRENT OF AREA table (table of CRAs),
– CURRENT OF RECORD table (table of CRRs) and
– CURRENT OF SET table (table of CRSs).

CURRENT OF AREA table
See currency table.

CURRENT OF RECORD table
See currency table.

CURRENT OF SET table
See currency table.

DAL
(Database Administrator Language) Comprises the commands which monitor and control a session.

data backup
Protection against loss of data as a result of hardware or software failure.

data deadlock
See deadlock.

data protection (privacy)
Protection against unauthorized access to data. Implemented in UDS/SQL by means of the schema/subschema concept and access authorization. Access rights are granted by means of the BPRIVACY utility routine.

database (DB)
Related data resources that are evaluated, processed and administered with the help of a database system.
A database is identified by the database name.
An UDS/SQL database consists of the user database and the compiler database.
To prevent the loss of data, a shadow database may be operated together with (i.e. parallel to) the original database.
database administrator
Person who manages and controls database operation. The DB administrator is responsible for the utility routines and the Database Administrator Language (DAL).

database copy
Copy of a consistent database; may be taken at a freely selectable point in time.

database compiler realm (DBCOM)
Stores information on the realms, records and sets defined by the user in the Schema DDL and Subschema DDL.

database copy update
Updating of a database copy to the status of a checkpoint by applying the appropriate after-images.

database directory (DBDIR)
Contains, among other things, the SIA, all the SSIAs and information on access rights.

database key (DB key)
Key whose value represents a unique identifier of a record in the database. It consists of the record reference number and the record sequence number. The database key values are either defined by the database programmer or automatically assigned by UDS/SQL.

database key item
Item of type DATABASE-KEY or DATABASE-KEY-LONG that is used for inserting database key values.
Items of type DATABASE-KEY and DATABASE-KEY-LONG differ in terms of the item length (4 bytes / 8 bytes) and value range.

DATABASE-KEY item
See database key item.

DATABASE-KEY-LONG item
See database key item.

database page
See page.

DATABASE-STATUS
Five-byte item indicating the database status and consisting of the statement code and the status code.
**database system**
Software system that supports all tasks in connection with managing and controlling large data resources. The database system provides mechanisms for stable and expandable data organization without redundancies. They allow many users to access *databases* concurrently and guarantee a consistent data repository.

**DB status file**
(database status file) Contains information on the most recently reset transactions. UTM-S or, in the case of distributed processing, UDS-D/openUTM-D needs this information for a session restart.

**DB configuration**
(database configuration) The databases attached to a DBH at any one point during session runtime. As the result of DAL commands or DBH error handling, the data base configuration can change in the course of a session. At the session start, the DB configuration may be empty. Databases can be attached with DAL commands after the start of the session. They can also be disconnected during the session with DAL commands.

**DBCOM**
See database compiler realm.

**DBDIR**
See database directory.

**DBH**
Database Handler: program (or group of programs) which controls access to the database(s) of a session and assumes all the attendant administrative functions.

**DBH end**
End of the DBH program run. DBH end can be either a session end or a session abort.

**DBH, independent**
See independent DBH.

**DB key**
See database key.

**DBH, linked-in**
See linked-in DBH.
DBH load parameters
See load parameters (DBH).

DBH start
Start of the DBH program run. DBH start can be either a session start or a session restart.

DBTT
(Database Key Translation Table) Table from which UDS/SQL can obtain the page address (act-key) of a record and associated tables by means of the database key value. The DBTT for the SSIA-RECORD consists only of the DBTT base. For all other record types, the DBTT consists of a base table (DBTT base) and possibly of one or more extension tables (DBTT extents) resulting from an online DBTT extension or created by BREORG.

DBTT anchor page
Page lying within the realm of the associated DBTT in which the DBTT base and DBTT extents are administered. Depending on the number of DBTT extents multiple chained DBTT anchor pages may be required for their administration.

DBTT base
see DBTT

DBTT extent
see DBTT

DBTT page
Page containing the DBTT or part of the DBTT for a particular record type.

DCAM
Component of the TRANSDATA data communication program.

DCAM application
Communication application using the DCAM communication method. A DCAM application enables communication between
– a DCAM application and terminals.
– different DCAM applications within the same or different hosts, and with remote configurations.
– a DCAM and a UTM application.

DDL
(Data Description Language) Formalized language for defining the logical data structure.
deadlock
Mutual blocking of transactions. A deadlock can occur in the following situations:
- Data deadlock: This occurs when transactions block each other with contending access.
- Task deadlock: This occurs when a transaction that is holding a lock cannot release it, since no UTM task is free. This deadlock situation can only occur with UDS/SQL-UTM interoperation.

descending key (DESC key)
Primary key of a set. Determines the sequence of member records in the set occurrences to reflect descending key values.

direct access
Access to a record via an item content. UDS/SQL supports direct access via the database key, hash routines and multi-level tables.

direct hash area
See hash area.

distributed database
A logically connected set of data resources that is distributed over more than one UDS/SQL configuration.

distributed transaction
Transaction that addresses at least one remote configuration. A transaction can be distributed over:
- UDS-D,
- openUTM-D,
- UDS-D and UTM-D.

distribution pool
Area in the independent DBH used for communication between UDSCT, server tasks, user tasks and the master task with regard to UDS-D-specific data. The distribution pool contains the distribution table and the UDS-D-specific system tables.
distribution table
Table created by UDS-D using the input file assigned in the distribution pool. With the aid of the distribution table, the distribution component in the user task decides whether a processing chain should be processed locally or remotely. Assigned in the distribution table are:
subschema - database
database - configuration
configuration - host computer.

DML
Data Manipulation Language: language for accessing a UDS/SQL database.

dummy subtransaction
A primary subtransaction is created by UDS-D when the first READY statement in a transaction addresses a remote database. A dummy subtransaction is used to inform the local configuration of the transaction so that the database can be recovered following an error.

duplicates header
Contains general information on a duplicates table or a page of a duplicates table, i.e.
– chaining reference to the next and previous overflow page
– the number of free bytes in the page of the duplicates table.

duplicates table
Special SEARCH-KEY table in which a key value which occurs more than once is stored only once. For each key value, the duplicates table contains:
– a table index entry with the key value and a pointer to the associated table entry
– a table entry (DB key list), which can extend over several pages, containing the record sequence numbers of the records which contain this key value.

duplicates table, main level
Main level, Level 0. Contains a table index entry and the beginning of the associated table entry (DB key list).

dynamic set
Set which exists only for the life of a transaction and which stores member records retrieved as result of search queries.

ESTIMATE-REPORT
Report produced after BGSIA run. Used to estimate the size of the user realms.
event name
Identification used in eventing.

exclusive buffer pool
Buffer which, in addition to the system buffer pools, is used exclusively for buffering pages of the specified database.

foreign key
Record element whose value matches the primary key values of another table (UDS/SQL record type). BPSQLSIA assigns foreign keys on the basis of set relationships.

FPA
See free place administration.

FPA base
See free place administration.

FPA extent
See free place administration.

FPA page
Free place administration page.

free place administration (FPA)
Free space is managed both at realm level (FPA pages) and at page and table level. Free place administration of the pages is carried out in a base table (FPA base) and possibly in one or more extension tables (FPA extents) created by means of an online realm extension or BREORG.

function code
Coding of a DML statement; included in information output by means of the DAL command DISPLAY or by UDSMON.

group item
Nameable grouping of record elements.

hash area
Storage area in which UDS/SQL stores data and from which it retrieves data on the basis of key values which are converted into relative page numbers. A hash area may contain the record addresses as well as the records themselves. A direct hash area contains the records themselves; an indirect hash area, by contrast, contains the addresses of records stored at some other location.
**hash routine**
Module which performs hashing.

**hashing**
Method of converting a key value into a page address.

**HASHLIB**
Module library for the storage of hash routines for one database.

**identifier**
Name allocated by the database designer to an item that UDS/SQL creates automatically. UDS/SQL adapts item type and length to the specified item usage.

**implicit set**
SYSTEM set created by UDS/SQL when a SEARCH key is defined at record type level.

**inconsistency**
State of the database in which the data values contained in it are inconsistent.

**independent DBH**
Independent program system enabling more than one user to access a single database (mono-DB operation) or several databases (multi-DB operation) simultaneously. The independent DBH is designed as a task family, consisting of
- a master task (UDSSQL)
- one or more server tasks (UDSSUB)
- an administrator task (UDSADM)

**index level**
Hierarchy level of an index page.

**index page**
Page in which the highest (lowest) key values of the next-lower level of an indexed table are stored.

**INDEX search key**
Secondary key. Used as access path for direct access via a multi-level table.

**indirect hash area**
See hash area.
integrity
State of the database in which the data contained in it is complete and free of errors.
– entity integrity
– referential integrity
– user integrity

interconfiguration
Concerning at least one remote configuration.

interconfiguration consistency
A distributed transaction that has caused updates in at least one remote configuration is terminated in such a way that the updates are either executed on the databases in each participating DB configuration or on none at all. Interconfiguration consistency is assured by the two-phase commit protocol.

interconfiguration deadlock
Situation where distributed transactions are mutually locked due to contending accesses.

interface
In software: memory area used by several different programs for the transfer of data.

internal version number
Each realm of the database, including DBDIR and DBCOM, has an internal version number which the utility routines (e.g. BREORG, BALTER) increment by one whenever a realm is updated. This internal version number is kept in the act-key-0 page of the realm itself and also in the PHYS VERSION RECORD in the DBDIR.

item
Smallest nameable unit of data within a record type. It is defined by item type and item length.

KDBS
Compatible database interface. Enables programs to be applied to applications of DB systems by different manufacturers.

key
Item used by the database programmer for direct access to records; an optimized access path is provided for the key by UDS/SQL in accordance with the schema definition.
**key, compound**
Key consisting of several *key items*.

**key item**
*Item* defined as a *key* in the *schema*.

**key reference number**
*Keys* are numbered consecutively in ascending order, beginning at 1.

**linked-in control system**
UDS/SQL component for *linked-in DBH*, responsible for control functions (corresponds to the *subcontrol system* of the *independent DBH*).

**linked-in DBH**
Module linked in to the current DB *application program* and controlling access to a single *database* (*mono-DB operation*) or several databases simultaneously (*multi-DB operation*).

**list**
Table containing the *member records* of a *set occurrence*. Used for *sequential* and *direct access* to member records.

**load parameters (DBH)**
Parameters requested by the *DBH* at the beginning of the *session*. They define the basic characteristics of a session.

**local application program**
An *application program* is local with regard to a *configuration* if it was linked to the configuration using `/SET-FILE-LINK LINK-NAME=DATABASE,FILE-NAME=conf-name`

**local configuration**
The *configuration* assigned to an *application program* before it is called using `/SET-FILE-LINK LINK-NAME=DATABASE,FILE-NAME=conf-name`. The application program communicates with the local configuration via the *communication pool*. The local configuration is in the same host as the application program.

**local database**
*Database* in a *local configuration*.

**local distribution table**
A *distribution table* is considered local to a *DBH* if it is held in the DBH’s *distribution pool*. 
local host
Host computer containing the application program.

local transaction
Transaction that only addresses the local configuration.

logging
Recording of all updates in the database.

logical connection
Assignment of two communication partners that enables them to exchange data. DCAM applications communicate via logical connections.

main reference
Transaction channel in the DBH.

mainref number
Number assigned to the transaction at READY. This number is unique only at a given time; at the end of the transaction, it is assigned to another transaction.

master task
Task of the independent DBH in which the UDSQL module executes. Controls the start and end of a session and communicates with the database administrator directly or via the administrator task.

member
See member record or member record type.

member, AUTOMATIC
Record is inserted at storage time.

member, MANDATORY
Record cannot be removed.

member, MANUAL
Record is not inserted automatically at storage time.

member, OPTIONAL
Record can be removed.

member record
Lower-ranking record in a set occurrence.
member record type
Lower-ranking record type in a set.

mono-DB configuration
Type of configuration where only one database takes part in a session.

mono-DB operation
Mode of database operation where the DBH uses only one database of a configuration.

multi-DB configuration
Type of configuration where several databases take part in a session.

multi-DB operation
Mode of database operation where the DBH uses several databases of a configuration.

multi-DB program
Application program that addresses more than one database. The databases may be part of one or more mono-DB or multi-DB configurations.

multi-level table
SEARCH KEY table which contains a line for each record of the associated record type or each member record of the set occurrence, as appropriate. Each line comprises the key value of the record and the record pointer. It is also referred to as an indexed table.

multithreading
A mechanism that enables the DBH to fully exploit the CPU. Multithreading means that the DBH processes several jobs concurrently by using so-called threads. Each thread has information on the current status of a particular job stored in it. When a job needs to wait for the completion of an I/O operation, DBH uses the CPU to process some other job.

network
All computers linked via TRANSDATA.

OLTP
(Online Transaction Processing) In an OLTP application, a very large number of users access the same programs and data. This usually occurs under the control of a transaction monitor (TP monitor).
online backup
If AFIM logging is active, the database can be saved during a session. The ability to save a database online is determined with the BMEND utility routine.

online DBTT extension
Automatic extension during ongoing database operation of the number of possible records of a record type. The DAL commands ACT DBTT-INCR, DEACT DBTT-INCR, DISPLAY DBTT-INCR and EXTEND DBTT can be used to administer the online extension of DBTTs.

online realm extension
Automatic extension of user realms and DBDIR in live database operation. For the purposes of administration of online extensibility of realms, you have the DAL commands ACT INCR, DEACT INCR, DISPLAY INCR and REACT INCR.

open transaction
Transaction which has not been closed with FINISH or FINISH WITH CANCEL, or with COMMIT or ROLLBACK.

openUTM
(universal transaction monitor) Facilitates the creation and operation of transaction-oriented applications.

operator task (OT)
See master task

original database
The term “original database” refers solely to the naming of the database files (dbname.dbfile), not to the status of the database content (see also shadow database).

overflow page
Page in hash areas and duplicates tables for storing data that does not fit in the primary page. Their structure is the same as that of the pages of the hash area or duplicates table in question.

owner
See owner record or owner record type.

owner record
Higher-ranking record in a set occurrence.

owner record type
Higher-ranking record type in a set.
page
Physical subunit of a *realm*. UDS/SQL identifies pages by means of unique keys *(act-key)*.
The length of a page may be optionally 2048, 4000 or 8096 bytes. All pages within a database must have the same length. Pages with a length of 4000 or 8096 bytes are embedded in a *page container*.

page address
In a page address, a distinction is made between the current address of a *page*, i.e. the *act-key*, and the probable address of a page, the *PPP*.

page container
Pages with a length of 4000 or 8096 bytes are embedded in a so-called page container, which consists of a 64-byte header that precedes the page and a 32-byte trailer at the end of the page.

page header (page info)
The first 20 bytes of a database *page* (except for the *FPA* and *DBTT* pages with a length of 2048 bytes). They contain:
- the *act-key* of the *page* itself,
- the number of *page index entries*
- the length and displacement of the bytes which are still vacant in this page.
- the *page type* (*ACT-Key-0 page*, *FPA page*, *DBTT page*, *DBTT anchor page*, normal data page or *CALC page*)

page index entry
Indicates the position of a *record* within a *page*.

page number
In each *realm* the *pages* are numbered consecutively in ascending order starting from 0. The page number is part of the *page address*.
Page number = PAM page number - 1 for databases with a page length of 2048 bytes
Page number = (PAM page number-1) / 2 for databases with a page length of 4000 bytes
Page number = (PAM page number-1) / 4 for databases with a page length of 8096 bytes.

password for UDS/SQL files
Password serving to protect the files created by UDS/SQL (default: C’UDS.’). The *DB administrator* can define other passwords with PP CATPASS or MODIFY-FILE-ATTRIBUTES.
pattern
Symbolic representation of all possible item contents, used at item definition.

pattern string
String defining a pattern.

PETA
Preliminary end of transaction: UDS-D or open UTM-D statement that causes a preliminary transaction end.
The PETA statement belongs to the first phase of the two-phase commit protocol which terminates a distributed transaction.
The PETA statement stores the following information failproof in the RLOG file of the local DBH:
– each updated page
– rollback and locking information
– the names of all participating configurations.
This information is required for any future warm start.

pointer array
Table of pointers to the member records of a set occurrence. Used for sequential and direct access to member records.

prepared to commit (PTC)
Part of the two-phase commit protocol:
State of a subtransaction after execution of a PETA statement and before receipt of the message that the complete transaction is to be terminated with FINISH or FINISH WITH CANCEL.

primary key
Distinguished from secondary keys for reasons of efficiency. Usually a unique identifier for a record.

primary key (SQL)
Record element uniquely identifying a record; output for UDS/SQL by BPSQLSIA.

primary subtransaction
Subtransaction that runs in the local configuration.
The primary subtransaction is opened by the first READY statement in a transaction on a local database.
If the first READY statement addresses a remote database, UDS-D generates a dummy subtransaction as the primary subtransaction.

PRIVACY-AND-IQF SCHEMA
UDS/SQL-internal schema for protection against unauthorized access.
PRIVACY-AND-IQF SUBSCHEMA
UDS/SQL-internal subschema for protection against unauthorized access.

probable position pointer
See PPP.

processing chain
Sequence of DML statements applied to a database within a transaction.

PPP (probable position pointer)
Probable address of a page, comprising realm number and page number.
UDS/SQL does not always update PPPs when the storage location of data is changed.

PTC state
See prepared to commit.

P1 eventing
Manner in which tasks communicate with each other.

READY
Start of a transaction or a processing chain in COBOL DML programs.

READYC
Start of a transaction or a processing chain in CALL DML programs.

realm
Nameable physical subunit of the database. Equivalent to a file. Apart from the user realms for user data there are also the realms DBDIR and DBCOM, which are required by UDS/SQL.

realm configuration
Comprises all the database realms taking part in a session.

realm copy
See database copy.

realm reference number
Realms are numbered consecutively in ascending order, starting with 1. The realm reference number (area reference) is part of the page address.
reconfiguration
Regrouping of databases in a DB configuration after a session abort. A prerequisite for reconfiguration is that the SLF has been deleted or that its contents have been marked as invalid.

record
Single occurrence of a record type; consists of one item content for each of the items defined for the record type and is the smallest unit of data managed by UDS/SQL via a unique identifier, the database key.

record address
Address of the page containing the record. See page address.

RECORD AREA
Area in the USER WORK AREA (UWA) which can be referenced by the user. The record area contains the record types and the implicitly defined items (IMPLICITLY-DEFINED-DATA-NAMES) of the database such as the AREA-ID items of the WITHIN clauses of the schema. The length of the record area is essentially defined by the record types contained in it.

record element
Item, vector or group item.

record hierarchy
Owner/member relationship between record types:
the owner record type is the higher-ranking part of the relationship;
the member record type is the lower-ranking part.

REC-REF
See record reference number.

record reference number
Record types are numbered consecutively in ascending order, starting at 1. The record reference number is part of the database key.

record SEARCH KEY table
SEARCH KEY table for selection of a record from a record type.

record sequence number (RSQ)
The record sequence number can be assigned by the database programmer; if not, UDS/SQL numbers the records of a record type contiguously in ascending order, in the sequence in which they are stored; numbering starts at 1. The record sequence number is part of the database key.
record type
Nameable grouping of record elements.

record type, linear
Record type that is neither the owner nor the member of a set (corresponds to record types of a conventional file).

referential integrity
Integrity of the relationships between tables (UDS/SQL record types).

remote application program
Application program that is not local with regard to a particular configuration.

remote configuration
DB-configurations that are not assigned to the application program via /SET-FILE-LINK LINK-NAME=DATABASE,FILE-NAME=conf-name but via the distribution table once the application program is running. The connection module of the application program communicates with the remote configurations via DCAM applications. Remote configurations can be situated on local or remote hosts.

remote database
Database in a remote configuration.

remote host
Host computer that is not local.

repeating group
Group item with repetition factor. The repetition factor, which must be greater than 1, specifies the number of duplicates of the group item to be incorporated in the repeating group.

request
The functions of the DAL commands ADD DB, ADD RN, DROP DB, DROP RN, NEW RLOG and CHECKPOINT are held in the DBH as "requests" and are not executed until the DAL command PERFORM is entered.

restart of BMEND
Resumption of an aborted BMEND run.

restart of a session
See session restart.
restructuring
Modification of the Schema DDL or SSL for databases already containing data.

return code
Internal code which the called program sends to the calling program; Return code $\neq 0$ means an error has occurred.

RLOG file
Backup file used by the DBH during a session to store before-images (BFIMs) and after-images (AFIMs) of data which is updated. With the aid of the RLOG file, the DBH can cancel updates effected by incomplete transactions. There is one RLOG file per configuration. An RLOG file consists of two physical files.

rollback
Canceling of all updates effected within a transaction.

RSQ
See record sequence number.

RUNUNIT-ID
See transaction identification.

Set Connection Data (SCD)
Linkage information for the records of a set occurrence.

schema
Formalized description of all data structures permitted in the database. A UDS/SQL schema is defined by means of the Schema DDL.

Schema DDL
Formalized language for defining a schema.

Schema Information Area (SIA)
The SIA contains the complete database definition. The DBH loads the SIA into main memory at the start of DB processing.

SEARCH KEY
Secondary key; access paths using secondary keys are created by UDS/SQL by means of hash routines and multi-level tables.

SEARCH KEY table
Multi-level table used by UDS/SQL as an access path via a secondary key.
secondary key
Any key which is not a primary key. Does not always identify a record uniquely, but is used for selecting a collection of records with the same key values.

secondary subtransactions
Subtransactions that address remote configurations.

sequence number
Identifier in the name of the ALOG files (000000001 - 999999999). The first ALOG file of a database is always numbered 000000001.

sequential access
Accessing a record on the basis of its position within a predefined record sequence.

server task
Task of the independent DBH in which the UDSSUB module executes; processes the requests of the DB application programs.

session
Period between starting and normal termination of the DBH (independent/linked-in) in which it is possible to work with the databases of the configuration. Normally, a session consists of a sequence of session segments and session interrupts.

session abort
Occurs when the DBH is terminated abnormally after a successful session start. A session abort can be caused by: power failure, computer failure, BS2000 problems, DBH problems, %TERM.

session end
Is the result of:
– DAL when using independent DBH,
– TERM in the DML application program when using linked-in DBH,
– DBH error handling.
During a session interrupt, the user can also effect session end by invalidating the SLF contents. Inconsistent databases can be made consistent again by a warm start, even without an SLF.

session interrupt
The period between a session abort and the related session restart.
Session Log File (SLF)
File which is permanently assigned to a session and which is required by the DBH in the event of a session restart. It contains information on the current DB configuration, the number of current file identifiers and the current values of the DBH load parameters.

session restart
Starting of the DBH, under the same configuration name and configuration ID, after a session abort. With the aid of the SLF, the DBH load parameters and the current file identifiers which existed when the session aborted are re-established, and the databases of the previous configuration are re-connected, if necessary by means of a warm start.

session segment
Period from the start of the DBH, either at the session start or a restart, to the normal session end or to a session abort.

session start
State of a session in which the DBH is started under a configuration name for which there is no Session Log File (SLF) with valid contents.

set
Nameable relationship between two record types.

set, dynamic
See dynamic set.

set, implicit
See implicit set.

set, singular
See SYSTEM set.

set, standard
See standard set.

set occurrence
Single instance of a set. Comprises exactly one owner record and any number of subordinate member records.

set reference number
Sets are numbered contiguously in ascending order, beginning at 1.
set SEARCH KEY table

SEARCH KEY table for selecting a member record from a set occurrence.

shadow database

Backup of all the files of a database, each saved under the name "dbname.dbfile.copyname".
A shadow database can be created at any time and processed parallel to the original database in RETRIEVAL mode.
In addition BMEND can be used to apply ALOG files that have already been closed to the database parallel to the UDS/SQL session.

Shared user buffer pool

Shared buffer of several databases which is used in addition to the System Buffer Pool, solely for buffering pages of the databases that have been assigned to it.

SIA

See Schema Information Area.

SIB

See SQL Interface Block.

SLF

See session log file.

sort key table

Table pointing to the member records of a set occurrence.

source program

Program written in a programming language and not yet translated into machine language.

spanned record

Record exceeding the length of a page. Only UDS/SQL-internal records can be spanned records;
User record types must not exceed
– 2020 bytes for a page length of 2048 bytes
– 3968 bytes for a page length of 4000 bytes
– 8064 bytes for a page length of 8096 bytes.

SQL

SQL is a relational database language which has been standardized by ISO (International Organization for Standardization).
SQL conversation
See conversation.

SQL DML
SQL Data Manipulation Language for querying and updating data.

SQL Interface Block (SIB)
Interface between UDS/SQL and SQL application program(s); contains the SQL statement, any existing parameters and the statement results.

SQL transaction
Related sequence of SQL statements which is processed by UDS/SQL either as a whole or not at all. This method ensures that the database(s) is/are always in a consistent state.

SSIA
See Subschema Information Area.

SSIA-RECORD
UDS/SQL-internal record type, located in the DBDIR. Records belonging to this type are, for example, the Schema Information Area (SIA) and the Subschema Information Areas (SSIAs).

SSITAB module
Module generated by the BCALLSLI utility routine; makes available the subschema information required by CALL DML programs.

SSL
See Storage Structure Language.

standard set
A set other than a dynamic, implicit or SYSTEM set.

statement code
Number stored in the first part of the DATABASE-STATUS item. Its function is to indicate which DML statement resulted in an exception condition.

status code
Number stored in the second part of the DATABASE-STATUS item. It indicates which exception condition has occurred.

Storage Structure Language (SSL)
Formalized language for describing the storage structure.
**string**

A series of consecutive alphanumerical characters.

**subcontrol system**

Component for the independent DBH. Responsible for control functions.

**subschema**

Section of a schema required for a particular application; it can be restructured, within limits, for the intended application; a subschema is defined by means of the Subschema DDL.

**Subschema DDL**

Formalized language for defining a subschema.

**Subschema Information Area (SSIA)**

The SSIA contains all subschema information required by the DBH to carry out, on behalf of the user, the database accesses permitted within the called subschema. The DBH loads the SSIA into main memory when it is referenced in a READY command.

**subschema module**

Module resulting from subschema compilation when a COBOL DML program is compiled. It must be linked in to the application program and includes the USER WORK AREA (UWA) as well as the RECORD AREA, which is also part of the base interface block (BIB). The name of the subschema module is the first 8 bytes of the subschema name.

**subschema record**

Record defined in the Subschema DDL.

**SUB-SCHEMA SECTION**

In COBOL programs with DML statements: section of the DATA DIVISION used for specifying the schema name and the subschema name.

**subtransaction**

In a distributed transaction, all the processing chains that address the databases in one configuration form a subtransaction.

**system area**

Realm required only by UDS/SQL. The system areas of a database include:
- the Database Directory (DBDIR),
- the Database Compiler Realm (DBCOM),
- the COBOL Subschema Directory (COSSD)
system break information
Indicates whether the database is consistent or inconsistent.

system buffer pools
Input/output buffer for database pages (see page). The buffer is part of the common pool (independent DBH) or the DBH work area (linked-in DBH). Its size is determined by the DBH load parameters 2KB-BUFFER-SIZE, 4KB-BUFFER-SIZE or 8KB-BUFFER-SIZE.

SYSTEM record
See anchor record.

SYSTEM set
Set whose owner record type is the symbolic record type SYSTEM.

table, multi-level
See multi-level table.

table (SQL)
A table in the context of SQL corresponds to a UDS/SQL record type.

table header
Contains general information on a table or table page:
- the table type and the level number of the table page,
- the number of reserved and current entries in this table page,
- the chaining reference to other table pages on the same level,
- the pointer to the associated table page on the next higher level,
- the pointer to the page containing the last table on the main level (for the highest-level table only).

table page
Page containing a table or part of a table.

TANGRAM
(Task and Group Affinity Management) Subsystem of BS2000/OSD that plans the allocation of processors for task groups which access large quantities of shared data in multi-task applications.
task attribute TP
There are 4 task attributes in BS2000/OSD: SYS, TP, DIALOG and BATCH. Special runtime parameters that are significant for task scheduling are assigned to each of these task attributes. In contrast to the other task attributes, the TP attribute is characterized by optimized main memory management that is specially tailored to transaction processing requirements.

task communication
Communication between the DBH modules. See also common pool.

task deadlock
See deadlock.

task priority
In BS2000/OSD, it is possible to define a priority for a task. This priority is taken into account when initiating and activating the task. Priorities may be fixed or variable. Variable priorities are Adapted dynamically; fixed priorities do not change. Note that UDS/SQL server tasks should be started with a fixed priority in order to ensure consistent performance.

TCUA
See Transaction Currency Area.

time acknowledgment
Message sent by the UDS-D task to the remote application program to indicate that there is still a DML statement being processed.

transaction (TA)
Related sequence of DML statements which is processed by UDS/SQL either as a whole or not at all. This method ensures that the database(s) is/are always in a consistent state. For UDS-D: The total set of subtransactions active at a given time.

transaction, committing a
Terminating a transaction with FINISH, i.e. all updates performed within the transaction are committed to the database.

transaction, rolling back a
Terminating a transaction with FINISH WITH CANCEL, i.e. all updates performed on the database within the transaction are rolled back.
Transaction Currency Area (TCUA)
Contains currency information.

transaction identification (TA-ID)
Assigned by the DBH to identify a particular transaction. Can be requested with the DAL command DISPLAY.

transfer pool
UDS-D-specific storage area in which the UDSCT receives the BIBs from remote application programs.

two-phase commit protocol
Procedure by which a distributed transaction that has made changes in at least one remote configuration is terminated in such a way as to safeguard inter-configuration consistency or UDS/SQL UTM-D consistency. The two-phase commit is controlled
– by the distribution component in the user task if the transaction is distributed via UDS-D.
– by UTM-D if the transaction is distributed via UTM-D or via UTM-D and UDS-D.

UDSADM
Module of the independent DBH; executes in the administrator task.

UDSHASH
Module generated by the BGSIA utility routine. It contains the names of all the hash routines defined in the Schema DDL.

UDSNET
Distribution component in the user task.

UDSSQL
Module of the independent DBH; executes in the master task.

UDSSUB
Module of the independent DBH; executes in the server task.

UDS-D task UDSCT
Task started for each configuration by UDS/SQL so that it can participate in distributed processing with UDS-D.
**UDS/UTM-D consistency**

A transaction that has updated both UTM data and UDS/SQL databases is terminated in such a way that the UTM data and the UDS/SQL databases are either updated together or not at all.

**unique throughout the network**

Unique in all the computers that are included in the network.

**user database**

The realms and files of the database required by the user in order to be able to store data in, and to retrieve data from a database are:

- the Database Directory (DBDIR),
- the user realms
- the module library for hash routines (HASHLIB).

**user realm**

A realm defined in the realm entry of the Schema DDL. It contains, among other things, the user records.

**user task**

Execution of an application program or UTM program, including the parts linked by the system.

**USER-WORK-AREA (UWA)**

Transfer area for communication between the application program and the DBH.

**UTM**

See openUTM.

**UWA**

See USER-WORK-AREA (UWA).

**vector**

Item with repetition factor. The repetition factor must be greater than 1. It specifies how many duplicates of the item are combined in the vector.

**version number, internal**

See internal version number.
warm start

A warm start is performed by UDS/SQL if an inconsistent database is attached to a session. For UDS/SQL this involves applying all updates of completed transactions to the database which have not yet been applied, rolling back all database transactions that are open, and making the database consistent. The related RLOG file and the DB status file are required for a warm start.
Abbreviations

ACS   Alias Catalog Service
Act-Key ACTual KEY
AFIM  AFter-IMage
AP    Application Program
ASC   ASCending
BIB   Base Interface Block
BFIM  BeFore-IMage
COBOL COmmon Business Oriented Language
CODASYL COnference on DAta SYstem Languages
CRA   CuRrent of Area
CRR   CuRrent of Record
CRS   CuRrent of Set
CRU   Current of RunUnit
COSSD COBOL SubSchema Directory
DAL   Database Administration Language
DB    DataBase
DBCOM DataBase COpiler Realm
DBDIR DataBase DIRectory
DBH   DataBase Handler
DB-Key DataBase KEY
DBTT  DataBase key Translation Table
DDL   Data Description Language
DESC  DESCending
DML   Data Manipulation Language
DRV   Dual Recording by Volume
DSA   Database System Access
DSSM  Dynamic SubSystem Management
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>FC</td>
<td>Function Code</td>
</tr>
<tr>
<td>FPA</td>
<td>Free Place Administration</td>
</tr>
<tr>
<td>GS</td>
<td>Global Storage</td>
</tr>
<tr>
<td>HSMS</td>
<td>Hierarchic Storage Management System</td>
</tr>
<tr>
<td>ID</td>
<td>IDentification</td>
</tr>
<tr>
<td>IMON</td>
<td>Installation Monitor</td>
</tr>
<tr>
<td>IQL</td>
<td>Interactive Query Language</td>
</tr>
<tr>
<td>IQS</td>
<td>Interactive Query System</td>
</tr>
<tr>
<td>KDBS</td>
<td>Kompatible Datenbank-Schnittstelle (= compatible database interface)</td>
</tr>
<tr>
<td>KDCS</td>
<td>Kompatible Datenkommunikationsschnittstelle (= compatible data communications interface)</td>
</tr>
<tr>
<td>LM</td>
<td>Lock Manager</td>
</tr>
<tr>
<td>LMS</td>
<td>Library Maintenance System</td>
</tr>
<tr>
<td>MPVS</td>
<td>Multiple Public Volume Set</td>
</tr>
<tr>
<td>MR-NR</td>
<td>MainRef NumbeR</td>
</tr>
<tr>
<td>MT</td>
<td>Master task</td>
</tr>
<tr>
<td>OLTP</td>
<td>OnLine transaction processing</td>
</tr>
<tr>
<td>openUTM</td>
<td>Universal Transaction Monitor</td>
</tr>
<tr>
<td>OT</td>
<td>Operator Task</td>
</tr>
<tr>
<td>PETA</td>
<td>Preliminary End of TrAnsaction</td>
</tr>
<tr>
<td>PPP</td>
<td>Probable Position Pointer</td>
</tr>
<tr>
<td>PTC</td>
<td>Prepared To Commit</td>
</tr>
<tr>
<td>PTT</td>
<td>Primäre Teiltransaktion (= primary subtransaction)</td>
</tr>
<tr>
<td>PVS</td>
<td>Public Volume Set</td>
</tr>
<tr>
<td>REC-REF</td>
<td>RE Cord REF erence number</td>
</tr>
<tr>
<td>RSQ</td>
<td>Record Sequence Number</td>
</tr>
<tr>
<td>SC</td>
<td>SubControl</td>
</tr>
<tr>
<td>SCD</td>
<td>Set Connection Data</td>
</tr>
<tr>
<td>SCI</td>
<td>Software Configuration Inventory</td>
</tr>
<tr>
<td>SECOLTP</td>
<td>SE Cure OnLine Transaction Processing</td>
</tr>
<tr>
<td>SECOS</td>
<td>SE cure COntrol System</td>
</tr>
<tr>
<td>SET-REF</td>
<td>SET-RE F erence</td>
</tr>
<tr>
<td>SIA</td>
<td>Schema Information Area</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>SIB</td>
<td>SQL Interface Block</td>
</tr>
<tr>
<td>SLF</td>
<td>Session Log File</td>
</tr>
<tr>
<td>SQL</td>
<td>Structured Query Language</td>
</tr>
<tr>
<td>SSD</td>
<td>Solid State Disk</td>
</tr>
<tr>
<td>SSIA</td>
<td>SubSchema Information Area</td>
</tr>
<tr>
<td>SSITAB</td>
<td>SubSchema Information TABle</td>
</tr>
<tr>
<td>SSL</td>
<td>Storage Structure Language</td>
</tr>
<tr>
<td>ST</td>
<td>ServerTask</td>
</tr>
<tr>
<td>STT</td>
<td>Sekundäre Teiltransaktion (= secondary subtransaction)</td>
</tr>
<tr>
<td>TA</td>
<td>TransAction</td>
</tr>
<tr>
<td>TA-ID</td>
<td>TransAction IDentification</td>
</tr>
<tr>
<td>TANGRAM</td>
<td>TAsk aNd GRoup Affinity Management</td>
</tr>
<tr>
<td>TCUA</td>
<td>Transaction CUrrency Area</td>
</tr>
<tr>
<td>UDS/SQL</td>
<td>Universal Database System/Structured Query Language</td>
</tr>
<tr>
<td>UWA</td>
<td>User Work Area</td>
</tr>
</tbody>
</table>
Related publications

The manuals are available as online manuals, see http://manuals.fujitsu-siemens.com, or in printed form which must be paid for and ordered separately at http://FSC-manualshop.com.

**UDS/SQL (BS2000/OSD)**
Application Programming
User Guide

*Target group*
This manual is intended for programmers of UDS/SQL database applications.

*Contents*
It describes the language concept and functionality of the DML, the transaction concept, the currency table, the functions of the COBOL- and CALL DML, the linking, loading and starting of a UDS/SQL-TIAM application program, the interoperation in a UDS/SQL-UTM application, the DMLTEST program and the DML status codes.

**UDS/SQL (BS2000/OSD)**
Creation and Restructuring
User Guide

*Target group*
This manual ist intended for the database administrator

*Contents*
It contains an overview of the files required by UDS/SQL. The manual also describes measures and utilities for creating, restructuring and converting a UDS/SQL database and for loading and unloading data.
Related publications

**UDS/SQL (BS2000/OSD)**
**Database Operation**
User Guide

*Target group*
The manual is intended for database administrators

*Contents*
This manual describes activities required for the administration and operation of a database. These include session administration and monitoring, database backup and recovery, and outputting database status values. Principles and operation of UDS-D are also described.

**UDS/SQL (BS2000/OSD)**
**Design and Definition**
User Guide

*Target group*
This manual is intended primarily for database designers. It also addresses programmers of UDS/SQL database applications and the database administrator.

*Contents*
It describes the phases in database design, the DDL data definition language, the SSL storage structure language, the subschema DDL and the relational schema.

**UDS/SQL (BS2000/OSD)**
**Messages**
User Guide

*Target group*
This manual is intended for database designers, database application programmers or the database administrator

*Contents*
All messages of the UDS/SQL Database System

**UDS/SQL (BS2000/OSD)**
**Recovery, Information and Reorganization**
User Guide

*Target group*
This manual is intended for the database administrator

*Contents*
This manual describes activities required for the administration and operation of a database. These include updating, reconstruction, consistency checking, outputting information, and reorganization.
**Related publications**

**UDS/SQL (BS2000/OSD)**

**Ready Reference**

*Target group*
UDS experts requiring practical information arranged for easy reference

*Contents*
All relevant syntax descriptions, tables and decision aids contained in the UDS/SQL manuals.

**UDS (BS2000)**

**Interactive Query System IQS**

**User’s Guide**

*Target group*
UDS users wishing to retrieve data from a database spontaneously and to make only minor changes

*Contents*
Commands enabling the user to access a UDS database without any previous programming

**UDS-KDBS (BS2000/OSD)**

**Compatible Database Interface**

**User Guide**

*Target group*
This manual is intended for the user of the compatible database interface with UDS/SQL.

*Contents*
The manual describes how to call and operate KDBS using the UDS/SQL database system.

**SQL for UDS/SQL**

**Language Reference Manual**

*Target group*
Programmers who want to access UDS databases using SQL statements.

*Contents*
SQL statements available for accessing UDS databases.
BS2000/OSD-BC
Commands, Volumes 1 - 5
User Guide

Target group
This manual is addressed to nonprivileged users and systems support staff.

Contents
Volumes 1 through 5 contain the BS2000/OSD commands ADD-... to WRITE-... (basic configuration and selected products) with the functionality for all privileges. The command and operand functions are described in detail, supported by examples to aid understanding. An introductory overview provides information on all the commands described in Volumes 1 through 5.
The Appendix of Volume 1 includes information on command input, conditional job variable expressions, system files, job switches, and device and volume types.
The Appendix of Volumes 4 and 5 contains an overview of the output columns of the SHOW commands of the component NDM. The Appendix of Volume 5 contains additionally an overview of all START commands.
There is a comprehensive index covering all entries for Volumes 1 through 5.

BS2000/OSD-BC
Commands, Volume 6, Output in S Variables and SDF-P-BASYS
User Guide

Target group
This manual is addressed to programmers and users who write procedures.

Contents
Volume 6 contains tables of all S variables that are supplied with values by the SHOW commands in conjunction with structured output. Further chapters deal with:
– introduction to working with S variables
– SDF-P-BASYS
Related publications

BS2000/OSD-BC
System Messages, Volumes 1 - 3
User Guide

Target group
This manual is addressed to systems support staff, operators and users.

Contents
Chapter 1 deals with message processing in BS2000/OSD. Chapter 2 contains the system messages of the message classes for the basic configuration of the BS2000/OSD operating system. The messages are arranged in alphabetical order by message class and are accompanied by explanatory texts where appropriate. Volume 2 and 3 contain the second and third part of system messages of the message classes for the basic configuration of the BS2000/OSD operating system.

BS2000/OSD-BC
Introductory Guide to Systems Support
User Guide

Target group
This manual is addressed to BS2000/OSD systems support staff and operators.

Contents
The manual covers the following topics relating to the management and monitoring of the BS2000/OSD basic configuration: system initialization, parameter service, job and task control, memory/device/system time/user/file/pubset management, assignment of privileges, accounting and operator functions.

BS2000/OSD-BC
Executive Macros
User Guide

Target group
The manual addresses all BS2000/OSD assembly language programmers.

Contents
The manual contains a summary of all Executive macros, detailed descriptions of each macro with notes and examples, including job variable macros, and a comprehensive general training section.
BS2000/OSD-BC
Introductory Guide to DMS
User Guide

Target group
This manual is addressed to nonprivileged users and systems support staff.

Contents
It describes file management and processing in BS2000.
Attention is focused on the following topics:
– volumes and files
– file and catalog management
– file and data protection
– OPEN, CLOSE and EOV processing
– DMS access methods (SAM, ISAM,...)

SDF (BS2000/OSD)
Introductory Guide to the SDF Dialog Interface
User Guide

Target group
BS2000/OSD users

Contents
This manual describes the interactive input of commands and statements in SDF format. A Getting Started chapter with easy-to-understand examples and further comprehensive examples facilitates use of SDF. SDF syntax files are discussed.

SORT (BS2000/OSD)
User Guide

Target group
This manual is intended for BS2000(OSD users and programmers.

Contents
It describes the principles, functions and statements for sorting and merging data records, calling the subroutine interface and the SORTZM access method. A chapter with examples instructs newcomers how to use SORT.
SPACEOPT (BS2000/OSD)  
Disk Optimization and Reorganization  
User Guide  

Target group  
This manual is addressed to systems support staff.  

Contents  
The SPACEOPT subsystem is used to optimize reorganization of the volumes on a pubset. Reorganization makes available large areas of contiguous free storage space and reduces the number of file extents. 
With the aid of SPACEOPT, users can assess the occupancy and fragmentation status of the volumes of a pubset as well as monitor the SPACEOPT jobs. 
Overview of contents:  
– SPACEOPT jobs and job options  
– assessment of the volume status  
– commands in alphabetical order  

LMS (BS2000)  
SDF Format  
User Guide  

Target group  
BS2000 users.  

Contents  
Description of the statements for creating and managing PLAM libraries and the members these contain. 
Frequent applications are illustrated with examples.  

DSSM/SSCM  
Subsystem Management in BS2000/OSD  
User Guide  

Target group  
This manual addresses systems support staff and software consultants of BS2000/OSD.  

Contents  
The following are described: BS2000/OSD subsystem concept, dynamic subsystem management (DSSM), subsystem catalog management (SSCM) and the associated commands and statements.
Related publications

**ARCHIVE (BS2000/OSD)**
User Guide

*Target group*
- BS2000/OSD users
- BS2000/OSD system administrators
- BS2000/OSD operators

*Contents*
Functions and statements of the program ARCHIVE for logical data saving. ARCHIVE is used for saving, reconstructing and transferring files and job variables.

**DRV (BS2000/OSD)**
*Dual Recording by Volume*
User Guide

*Target group*
Systems support, operators and nonprivileged users

*Contents*
The manual describes the recording method DRV (Dual Recording by Volume) which allows data to be stored in duplicate on two disks. The use of DRV in the computer center increases the availability of the data stored on disk. The manual contains descriptions of all the procedures, commands and macro extensions required for the installation, use, control and monitoring of DRV. The use of DRV to migrate disks is also described in detail.

**HSMS / HSMS-SV (BS2000/OSD)**
*Hierarchical Storage Management System*
*Volume 1: Functions, Management and Installation*
User Guide

*Target group*
- BS2000/OSD users
- BS2000/OSD system administrators
- HSMS administrators

*Contents*
- Description of the data saving, archival, migration and data transfer functions
- HSMS management, invocation, execution and installation
- HSMS messages
Related publications

**SECOS (BS2000/OSD)**
Security Control System
User Guide

*Target group*
- BS2000 system administrators
- BS2000 users working with extended access protection for files

*Contents*
Capabilities and application of the functional units:
- SRPM (System Resources and Privileges Management)
- SECOS-KRB (Authentication with Kerberos)
- SRPMSSO (Single Sign On)
- GUARDS (Generally Usable Access Control Administration System)
- GUARDDEF (Default Protection)
- GUARDCOO (Co-owner Protection)
- SAT (Security Audit Trail).

**openNet Server (BS2000/OSD)**
BCAM
Reference Manual

*Target group*
The manual is intended for network operators, generators and administrators who define BS2000 systems.

*Contents*
BCAM Volume 2 is based on Volume 1 and describes in detail the BCAM commands required for generation and operation. The KOGS macros required for static generation are introduced and the BCAM messages are listed.

**DCAM (BS2000/OSD, TRANSDATA)**
Program Interfaces
Reference Manual

*Target group*
- Managers
- Application planners
- Programmers
- System and network administrators

*Contents*
Description of the Data Communication Access Method DCAM
Related publications

**DCAM (BS2000/OSD, TRANSDATA)**

*Macros User Guide*

*Target group*

Programmers of DCAM ASSEMBLER programs

*Contents*

– Special techniques when using DCAM macros
– DCAM macros, arranged according to functions
– Catalog of all DCAM macros

**TransView-NMA/-NMAE, TransView-NTAC2, NTAC2E (TRANSDATA, BS2000)**

*Network Management in BS2000 User Guide*

*Target group*

The manual addresses network planners, network administrators, network operators, and maintenance and diagnostics engineers.

*Contents*

It deals with network management for BS2000 systems. It describes how these products are used, their mode of operation and their interaction with other products. The manual is task-oriented; the complete reference for all the network management commands is no longer contained here, but in the “Network Management Commands” manual.

**OMNIS/OMNIS-MENU (TRANSDATA, BS2000/OSD)**

*Functions and Commands User Guide*

*Target group*

– OMNIS administrators
– OMNIS users

*Contents*

– Essential information for the use of OMNIS
– Description of the functions provided and possible implementations of OMNIS
OMNIS/OMNIS-MENU (TRANSDATA, BS2000)
Administration and Programming
User Guide

Target group

- OMNIS administrators
- Programmers

Contents
Introduction to OMNIS administration, the OMNIS utility routines and the application interface for extending the OMNIS functionality

openUTM
Concepts and Functions
User Guide

Target group
Anyone who wants information about the functionality and performance capability of openUTM.

Contents
The manual contains a general description of all the functions and features of openUTM, plus introductory information designed to help first-time users of openUTM.

openUTM (BS2000/OSD, UNIX, Windows)
Programming Applications with KDCS for COBOL, C and C++
User Guide

Target group
This manual is intended for programmers who wish to use the KDCS program interface for programming UTM applications.

Contents
The manual describes the KDCS interface in the form valid for COBOL, C and C++. This interface incorporates both the basic functions of the Universal Transaction Monitor and the calls for distributed processing. It also contains a description of working together with databases.
Related publications

**openUTM (BS2000/OSD, UNIX, Windows)**
**Generating Applications**
User Guide

*Target group*
This manual is designed for use by application planners and developers as well as operators of UTM applications.

*Contents*
This manual describes how to define the configuration for a UTM application using the UTM tool KDCDEF and how to create the KDCFILE. One chapter also goes into more detail about the generation of selected objects and functions of the application.

Additional topics include the dynamic configuration of an application and the updating of the KDCFILE using the tool KDCUPD.

**openUTM (BS2000/OSD, UNIX, Windows)**
**Administering Applications**
User Guide

*Target group*
This manual is intended for everyone responsible for administering openUTM applications and generating administration programs.

*Contents*
The manual describes the program interface to administration, which enables you to generate your own administration programs. It also describes the command interface to administration and the options available for the administration of message queues and printers.

**openUTM (BS2000/OSD)**
**Messages, Debugging and Diagnostics**
User Guide

*Target group*
This manual is intended for programmers, generators and administrators of UTM applications in BS2000/OSD.

*Contents*
The manual describes the debugging of UTM applications, the format of UTM dumps, behavior in the event of errors, and the openUTM message concept. It also includes all messages and return codes output by openUTM.
COBOL2000 (BS2000/OSD)
COBOL Compiler
Reference Manual

Target group
COBOL users in BS2000/OSD

Contents
– COBOL glossary
– Introduction to Standard COBOL
– Description of the full language set of the COBOL2000 compiler:
  formats, rules and examples illustrating the COBOL ANS85 language elements of the
  "High" language subset, the Fujitsu Siemens-specific extensions and the extensions
  defined by the forthcoming COBOL standard, specifically the object orientation.

COBOL2000 (BS2000/OSD))
COBOL Compiler
User’s Guide

Target group
COBOL users of BS2000/OSD

Contents
– Using the COBOL2000 compiler
– Linking, loading and starting of COBOL programs
– Debugging aids
– File processing with COBOL programs
– Checkpointing and restart
– Program linkage
– COBOL2000 and POSIX
– Useful software for COBOL users
– Messages of the COBOL2000 system
Related publications

COBOL85 (BS2000)
COBOL Compiler
Reference Manual

Target group
COBOL users in BS2000

Contents
– COBOL glossary
– Introduction to Standard COBOL
– Description of the full language set of the COBOL85 compiler: formats, rules and examples illustrating the COBOL ANS85 language elements of the "High" language subset, and the Siemens Nixdorf-specific extensions.

COBOL85 (BS2000)
COBOL Compiler
User's Guide

Target group
COBOL users of BS2000

Contents
– Generation of the COBOL85 compiler and the software required for the linking, loading and debugging of COBOL programs
– File processing with COBOL programs
– Inter-program communication
– Structure of the COBOL85 system
– Compiler messages and runtime system messages
Related publications

**DRIVE/WINDOWS** (BS2000)
Programming System
User Guide

*Target group*
Application programmers

*Contents*
– Introduction to the programming system DRIVE/WINDOWS
– Explanation of the functions available in interactive mode
– Installation
– DRIVE/WINDOWS generation and administration

**DRIVE/WINDOWS** (BS2000)
Programming Language
Reference Guide

*Target group*
Application programmers

*Contents*
Description of program creation including alpha screen forms, as well as the use of DRIVE list forms and the report generator.

**DRIVE/WINDOWS** (BS2000)
System Directory of DRIVE Statements
Reference Manual

*Target group*
Applications programmers

*Contents*
Syntax and range of functions of all DRIVE statements. DRIVE messages and keywords.

**DRIVE/WINDOWS** (BS2000/SINIX)
Directory of DRIVE SQL Statements for UDS
Reference Manual

*Target group*
Application programmers

*Contents*
A concise description of the syntax and scope of functions of all the DRIVE SQL statements for UDS.
Related publications

**DAB (BS2000/OSD)**
**Disk Access Buffer**
User Guide

*Target group*
This manual is addressed to systems support.

*Contents*
The manual begins with some introductory chapters dealing with DAB caching, the DAB cache media and the DAB functions, and continues with detailed descriptions of the DAB commands.

Overview of contents:
- DAB caching, DAB media, DAB functions
- DAB application notes, performance behavior, installation, starting and terminating DAB
- DAB commands and messages
- DAB error recovery

**BS2000/OSD**
**Softbooks English**
CD-ROM

*Target group*
BS2000/OSD users

*Contents*
The CD-ROM "BS2000/OSD SoftBooks English" contains almost all of the English manuals and README files for the BS2000 system software of the latest BS2000/OSD version and also of the previous versions, including the manuals listed here. These Softbooks can also be found in the Internet on our manual server. You can browse in any of these manuals or download the entire manual.

*Order number*
U26175-J8-Z125-1-76

*Internet address*
http://manuals.fujitsu-siemens.com
Index

A
access
  contending 283
direct 283
hash area 199
relational 161
sequential 283
access authorization 283
access path 283
access rights 283
access type 283
act-key 283
act-key-0 page 284
  check 58
  print 218
act-key-N page 284
additional link
  member records 138
ADD-REALM statement
  BMEND 26
address
  hash area 134, 143
  physical 284
  table 141
administrator task 284
AFIM 284
  apply 47
AFIM logging 243
after-image 284
  ALOG file 284
  RLOG file 284
after-image file 243
ALIAS item
  length 157
allocate buffer size 27
ALLOCATE-BUFFER-POOL statement
  BMEND 27
  BREORG 245
ALOG file 284
ALOG sequence number 284
alog-seq-no 16
anchor record 138, 285
  database key 138
appl 16
application 285
application program (AP) 285
apply AFIMs 47
area 285
AREA INFORMATION 126, 147
ascending key (ASC key) 285
attach realm 26
authorization 285
authorized users 285

B
backup database 285
Base Interface Block (BIB) 285
base tables
  SQL accesses 171
BCHECK 57
  CHECK statement 71
  command sequence 82
  consistency criterion 72
  error messages 90
  examples 89
  execution messages 90
  inconsistency messages 94
  interactive input of statements 69
  job switch 115
  KEY statement 80
Index

BCHECK (cont.)
  messages 89
  REALM statement 75
  RECORD statement 76
  SCHEMA statement 74
  SCRTCH1 file 66
  SET statement 78
  SORTCORE statement 70
  SORTWK file 66
  system environment 61, 62
  TYPE statement 72
  warnings 89
  work files 66
bcheck inconsistencies 57
before-image 285
BFIM 285
BIB (Base Interface Block) 285
BMEND
  ADD-REALM statement 26
  ALLOCATE-BUFFER-POOL statement 27
command sequence 50
  DISABLE-ONLINE-COPY statement 28
  ENABLE-ONLINE-COPY statement 29
  END statement 30
  functions 22
  job variables 51
  KILL-LOG statement 31
  OPEN-DATABASE statement 32
  REMOVE-REALM statement 33
  SHOW-LOG-INFORMATION statement 34
  START-LOG statement 40
  STOP-LOG statement 45
  UNDO statement 46
  UPDATE-DATABASE statement 47
BMODTT 277
command sequence 281
  KEEP statement 279
  RESET statement 280
  REUSE statement 279
  SET statement 280
  statements 279
  system environment 278
BPRECED 205
  command sequence 234
  DISPLAY CALC statement 225
  DISPLAY DATA statement 230
  DISPLAY DBTT statement 223
  DISPLAY PAGE ZERO statement 218
  DISPLAY-FPA statement 222
  PRINT statement 215
  REALM statement 214
  SCHEMA statement 213
  statement sequence 211
  statements 211
  system environment 208
BPSIA 117
  command sequence 121
  DISPLAY-Schema statement 119
  DISPLAY-SUBSCHEMA statement 119, 120, 203, 233
  statements 119
  system environment 118
BPSQLSIA
  command sequence 167
  conversion rules 169
  END statement 164
  OPEN-DATABASE statement 165
  output 168
  PRINT-RELATIONAL-SCHMAINFO statement 166
  system environment 162
  utility routine 161
BREORG 236
  ALLOCATE-BUFFER-POOL statement 245
  command sequence 271
  END statement 246
  examples 273
  functions 236
  MODIFY-REALM-SIZE statement 247
  MODIFY-RECORD-POPULATION statement 250
  OPEN-DATABASE statement 253
  REORGANIZE-CALC statement 254
  REORGANIZE-POINTERS statement 261
  REORGANIZE-SET statement 262
  SPECIFY-SCHEMA statement 268
BREORG (cont.)
SPECIFY-SUBSCHEMA statement 269
system environment 237
task switch 29 243
UNDO statement 270
work files 239
BSTATUS 178
command sequence 204
DISPLAY CALC statement 197
DISPLAY RECORD statement 194
DISPLAY RECORDNUMBER statement 201
DISPLAY TABLE FOR OWNER statement 192
DISPLAY TABLE FOR SET statement 189
DISPLAY-REALM statement 186
SUBSCHEMA statement 185
system environment 181
work files 182
buffer pools
see exclusive buffer pool
see system buffer pools
buffer size
allocate 27
define 245
C
CALC areas
reorganize 254, 275
CALC INFORMATION 133
CALC key 133, 142, 150, 154, 157, 197, 198, 210, 286
length 133, 142, 150, 157
statistics 180, 197
table 142, 228
CALC KEY INFORMATION 150
CALC key statistics
print 197
CALC page 228, 286
check 58
print 225
CALC SEARCH key 225, 286
CALC table 286
CALC-SEARCH-KEY-INFORMATION 142
calculate size
SORTWK file 183
CALL DML 286
catalog identifier 286
catid 16
chain 286
CharacterSeparatedValues(CSV) 286
check
act-key-0 page 58
CALC pages 58
coherence 60, 61
compiler database 57
consistency of database 57
consistency, physical 57
database, original 60
for consistency 57
original database 60
page header 58
page structures of realms 58
physical structures 57
PRIVACY-AND-IQF database 57
record displacement 58
record sequence numbers 58
shadow database 60
sort sequence of keys 58
system data 57
table pages 58
user realm 57
check records 286
generate 58
check run
database files 62
offline 61
online 61
parallel 61
CHECK statement
BCHECK 71
check system data 57
checking
incremental 60
overall 60
checking mode
select 71
set 58
Index

checking procedures 59
checkpoint 286
CHECK-TABLE 287
COBOL DML 287
COBOL runtime system 287
COBOL Subschema Directory (COSSD) 287
CODASYL subschema 161
coherence check 60, 61
column number
  DBTT 141, 191, 193
command sequence
  BCHECK 82
  BMEND 50
  BMODTT 281
  BPRECORD 234
  BPSIA 121
  BPSQLSIA 167
  BREORG 271
  BSTATUS 204
common memory 287
common pool 287
communication partners 287
communication pool 287
compatible database interface 287, 298
compiler data base 287
compiler database
  check 57
  COMPILER-SCHEMA 288
  COMPILER-SUBSCHEMA 288
compound key 288
compression 129, 288
configuration 288
configuration identification 288
configuration name 288
connection module 288
consistency 288
  check for 57
  logical 288
  of database, check 57
  physical 289
  physical, checking 57
  point 289
  record 289
  storage 289
  consistency criterion
    BCHECK 72
    select 72
contending access 289
conversation 289
conversion rules
  BPSQLSIA 169
copy 289
copy-name 16
COSSD 289
counter procedure 59
CRA 289
creation date
  realm 220
CRR 289
c-string 16
CSV 290
CSV format
  BPRECORD 218, 222, 223, 225, 230
  BPSIA 119, 120
  BSTATUS 186, 189, 192, 194, 197, 201
  output 119, 120, 186, 189, 192, 194, 197, 201
  218, 222, 223, 225, 230
  csv-dateiname 16
currency table 147, 290
  length 145
CURRENT
  OF AREA table 290
  OF RECORD table 290
  OF SET table 290
Current
  Record of Rununit 290
  Record of Set (CRS) 289
D
dal-cmd 16
data backup 290
data deadlock 290
Data Manipulation Language (DML) 295
data pages 230
  print 230
data protection (privacy) 290
data types 16
Index

database
  open 32, 165, 253
database (DB) 290
database administrator 291
Database Administrator Language (DAL) 290
database compiler realm (DBCOM) 291
database copy 291
database copy update 291
database directory (DBDIR) 291
database key 291
  anchor record 138
  item 149, 291
database operation
  SHARED-RETRIEVAL 61
database page 291
database pages 126
  length 124, 219
  occupancy 187
database system 292
database, original
  check 60
DATABASE-KEY item 291
DATABASE-KEY-LONG item 291
DATABASE-STATUS 291
date 16
DB configuration 292
DB key 148, 292
DB status file 292
DBCOM 292
DBDIR 292
DBH 292
  end 292
  independent 292
  linked-in 292
  start 293
DBH load parameters 293
dbname 16
DBTT 293
  column number 141, 191, 193
  filling ratio 196
  list entries 223
  page 293
  size 195
  anchor page 293
  anchor table 132
  base 293
  entries
    description 223
    length 132
    used 195
  extent 293
  extents
    number 132, 196
  INFORMATION 131
  DCAM 293
  DCAM application 293
  DDL 293
deactivate logging 45
deadlock 294
define
  buffer size 245
  output scope 215
define size
  sort buffer 70
descending key (DESC key) 294
description
  DBTT entries 223
  realm statistics 187
  table 233
designate
  schema 213
  subschema 185
determine size
  hash area 258
device 16
direct access 294
direct hash area 294
disable
  online-copy facility 28
DISABLE-ONLINE-COPY statement
  BMEND 28
DISPLAY CALC statement
  BPRECORD 225
  BSTATUS 197
DISPLAY DATA statement
  BPRECORD 230
DISPLAY DBTT statement
  BPRECORD 223
DISPLAY PAGE ZERO statement  
  BPRECORD  218  
DISPLAY RECORD statement  
  BSTATUS  194  
DISPLAY RECORDNUMBER statement  
  BSTATUS  201  
DISPLAY statement  
  output in CSV format  119, 120, 186, 189, 192, 194, 197, 201, 218, 222, 223, 225, 230  
DISPLAY TABLE FOR OWNER statement  192  
  BSTATUS  192  
DISPLAY TABLE FOR SET statement  
  BSTATUS  189  
DISPLAY-FPA statement  
  BPRECORD  222  
DISPLAY-REALM statement  
  BSTATUS  186  
DISPLAY-SCHEMA statement  
  BPSIA  119  
DISPLAY-SUBSCHEMA statement  
  BPSIA  119, 120  
distributed database  294  
distributed transaction  294  
distribution pool  294  
distribution table  295  
DRIVE/SQL  161  
dummy subtransaction  295  
duplicates header  295  
duplicates table  233, 295  
  main level  295  
dynamic set  136, 137, 295  
E  
enable online-copy facility  29  
ENABLE-ONLINE-COPY statement  
  BMEND  29  
END statement  
  BMEND  30  
  BPSIA  120, 233  
  BPSQLSIA  164  
  BREORG  246  
  BSTATUS  203  
entries  
  table  232  
error messages  
  BCHECK  90  
ESQL  161  
ESTIMATE-REPORT  295  
event name  296  
extamples  
  BCHECK  89  
  BREORG  273  
exclusive buffer pool  296  
exection messages  
  BCHECK  90  
EXTENSIBILITY  196  
extensibility  
  extensibility  196  
extensibility, record type  196  
F  
filling percentage  
  hash area  199  
  pages  187  
filling ratio  
  DBTT  196  
  set tables  191, 193  
  table  191, 193  
foreign key  296  
format subschema  150, 151  
FPA  296  
FPA base  219, 296  
FPA entries  
  list  222  
FPA extent  126, 219, 247, 296  
FPA page  296  
free pages  187  
free place administration  124, 126, 223, 247, 296  
free space  
  hash area  199  
function code  296  
functions  
  BREORG  236
Index

G
generate check records  58
group item  296

H
hash area  198, 296
   access  199
   address  134, 143
determine size  258
filling percentage  199
free space  199
number of overflow pages  199
number of primary pages  143, 198
occupancy level  199
print  197, 225, 228
hash routine  297
hashing  297
HASHLIB  297
header table  232
host  17

I
identifier  297
identify schema  74
implicit set  129, 136, 297
inconsistency  297
   BCHECK  57
inconsistency messages
   BCHECK  94
incremental checking  60, 61
independent DBH  297
index level  297
index page  297
INDEX search key  297
indirect hash area  297
input
   terminate  30, 164
termination  246
integer  17
integrity  298
interactive input of statements
   BCHECK  69
interconfiguration  298
   consistency  298
deadlock  298
interface  298
internal realm version  220
internal version number  298
item  298
   database key  149
   length  159
   variable  152
item string  152
   length  152
ITEM STRING LIST  151
item type  150, 157, 159

J
job switch BCHECK  115
job variable
   BMEND  51
   structure  52
   update  55

K
KDBS  287, 298
KEEP statement
   BMODTT  279
key  298
   compound  299
   length  123, 133, 142, 154, 157
   number  123, 140, 142, 146, 154, 159, 228
KEY INFORMATION  139, 158
key item  299
   length  140, 154, 159
   number  159
type  159
KEY ITEM LIST  153
key reference number  299
KEY statement
   BCHECK  80
keys
   number per record type  148
   number per set  156
   subschema  158
keyword  12
KILL-LOG statement
   BMEND  31
kset    17

L
length
   ALIAS item    157
   CALC key  142, 150, 157
   CALC keys    133
   currency table    145
   database pages 124, 219
   DBTT entries    132
   item    159
   item string    152
   key  123, 133, 142, 154, 157
   key item  140, 154, 159
   record type    123, 129
   set connection data    138
   SSIA    144
linked-in control system    299
linked-in DBH    299
list    299
   FPA entries    222
list entries
   DBTT    223
load parameters DBH    299
local application program    299
local configuration    299
local database    299
local distribution table    299
local host    300
local transaction    300
LOCATION MODE    129, 156
logging    300
   deactivate    45
   start    40
   stop    31
logging information
   output    34
logical connection    300

M
main reference    300
mainref number    300
master task    300
member    300
   AUTOMATIC    300
   MANDATORY    300
   MANUAL    300
   OPTIONAL    300
member record    300
   additional link    138
member record type    138, 301
membership
   set    136
messages
   BCHECK    89
Messages BCHECK    89
modify
   realm size    247
   record population    250
MODIFY-REALM-SIZE statement
   BREORG    247
MODIFY-RECORD-POPULATION statement
   BREORG    250
mono-DB configuration    301
mono-DB operation    301
multi-DB configuration    301
multi-DB operation    301
multi-DB program    301
multi-level table    301
multithreading    301

N
name    17
   realm    187, 214
   record type    129, 203, 228
   schema    121, 122, 213
   set    136, 228
   subschema    121, 144
network    301
notational conventions    12
   SDF statements    14
<table>
<thead>
<tr>
<th>Term</th>
<th>Page Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>number</td>
<td></td>
</tr>
<tr>
<td>DBTT extents</td>
<td>132, 196</td>
</tr>
<tr>
<td>key</td>
<td>123, 140, 142, 146, 154, 159, 228</td>
</tr>
<tr>
<td>key items</td>
<td>159</td>
</tr>
<tr>
<td>of DBTT entries</td>
<td>132</td>
</tr>
<tr>
<td>of records stored</td>
<td>203</td>
</tr>
<tr>
<td>overflow pages</td>
<td>199</td>
</tr>
<tr>
<td>primary pages</td>
<td>134</td>
</tr>
<tr>
<td>realm</td>
<td>123, 125, 126, 132, 141, 145, 149, 203, 210, 219</td>
</tr>
<tr>
<td>record type</td>
<td>123, 128, 129, 132, 228</td>
</tr>
<tr>
<td>set</td>
<td>123, 136, 139, 142, 158</td>
</tr>
<tr>
<td>stored records</td>
<td>191, 195</td>
</tr>
<tr>
<td>number of index levels</td>
<td></td>
</tr>
<tr>
<td>set tables</td>
<td>191</td>
</tr>
<tr>
<td>tables</td>
<td>191, 194</td>
</tr>
<tr>
<td>number of overflow pages</td>
<td></td>
</tr>
<tr>
<td>hash area</td>
<td>199</td>
</tr>
<tr>
<td>number of primary pages</td>
<td></td>
</tr>
<tr>
<td>hash area</td>
<td>143, 198</td>
</tr>
<tr>
<td>number per realm</td>
<td></td>
</tr>
<tr>
<td>record type</td>
<td>128</td>
</tr>
<tr>
<td>number per record type</td>
<td></td>
</tr>
<tr>
<td>keys</td>
<td>148</td>
</tr>
<tr>
<td>number per set</td>
<td></td>
</tr>
<tr>
<td>keys</td>
<td>156</td>
</tr>
<tr>
<td>number per subschema</td>
<td></td>
</tr>
<tr>
<td>realm</td>
<td>145</td>
</tr>
<tr>
<td>occupancy</td>
<td></td>
</tr>
<tr>
<td>database pages</td>
<td>187</td>
</tr>
<tr>
<td>occupancy level</td>
<td></td>
</tr>
<tr>
<td>hash area</td>
<td>199</td>
</tr>
<tr>
<td>overflow pages</td>
<td>199</td>
</tr>
<tr>
<td>offline check run</td>
<td>61</td>
</tr>
<tr>
<td>OLTP</td>
<td>301</td>
</tr>
<tr>
<td>online backup</td>
<td>302</td>
</tr>
<tr>
<td>online check run</td>
<td>61</td>
</tr>
<tr>
<td>online DBTT extension</td>
<td>302</td>
</tr>
<tr>
<td>online realm extension</td>
<td>302</td>
</tr>
<tr>
<td>online-copy facility</td>
<td></td>
</tr>
<tr>
<td>disable</td>
<td>28</td>
</tr>
<tr>
<td>enable</td>
<td>29</td>
</tr>
<tr>
<td>open database</td>
<td>32, 165, 253</td>
</tr>
<tr>
<td>open transaction</td>
<td>302</td>
</tr>
<tr>
<td>OPEN-DATABASE statement</td>
<td></td>
</tr>
<tr>
<td>BMEND</td>
<td>32</td>
</tr>
<tr>
<td>BPSQLSIA</td>
<td>165</td>
</tr>
<tr>
<td>BREORG</td>
<td>253</td>
</tr>
<tr>
<td>openUTM</td>
<td>302</td>
</tr>
<tr>
<td>operator task (OT)</td>
<td>302</td>
</tr>
<tr>
<td>optional word</td>
<td>12</td>
</tr>
<tr>
<td>original database</td>
<td>302</td>
</tr>
<tr>
<td>check</td>
<td>60</td>
</tr>
<tr>
<td>original realms</td>
<td></td>
</tr>
<tr>
<td>overall checking</td>
<td>62</td>
</tr>
<tr>
<td>output</td>
<td></td>
</tr>
<tr>
<td>BPSQLSIA</td>
<td>168</td>
</tr>
<tr>
<td>in CSV format</td>
<td>119, 120, 186, 189, 192, 194, 197, 201, 218, 222, 223, 225, 230</td>
</tr>
<tr>
<td>logging information</td>
<td>34</td>
</tr>
<tr>
<td>output scope</td>
<td></td>
</tr>
<tr>
<td>define</td>
<td>215</td>
</tr>
<tr>
<td>overall checking</td>
<td>60, 61</td>
</tr>
<tr>
<td>original realms</td>
<td>62</td>
</tr>
<tr>
<td>shadow database</td>
<td>63</td>
</tr>
<tr>
<td>overflow pages</td>
<td>302</td>
</tr>
<tr>
<td>number</td>
<td>199</td>
</tr>
<tr>
<td>occupancy level</td>
<td>199</td>
</tr>
<tr>
<td>remove</td>
<td>256, 257</td>
</tr>
<tr>
<td>owner</td>
<td>302</td>
</tr>
<tr>
<td>statistics</td>
<td>179, 193</td>
</tr>
<tr>
<td>owner record</td>
<td>302</td>
</tr>
<tr>
<td>owner record type</td>
<td>138, 302</td>
</tr>
<tr>
<td>owner statistics</td>
<td></td>
</tr>
<tr>
<td>print</td>
<td>192</td>
</tr>
<tr>
<td>P1 eventing</td>
<td>305</td>
</tr>
<tr>
<td>page</td>
<td>303</td>
</tr>
<tr>
<td>page address</td>
<td>303</td>
</tr>
<tr>
<td>page container</td>
<td>303</td>
</tr>
<tr>
<td>page header</td>
<td>216, 217, 228</td>
</tr>
<tr>
<td>check</td>
<td>58</td>
</tr>
<tr>
<td>page header (page info)</td>
<td>303</td>
</tr>
<tr>
<td>PAGE INDEX</td>
<td>217, 228, 232</td>
</tr>
<tr>
<td>page index entry</td>
<td>215, 216, 217, 228, 303</td>
</tr>
</tbody>
</table>
Index

PAGE INFO 217, 228, 232
page number   126, 132, 138, 212, 223, 233, 303
page numbers  222
page selection 212
page structures of realms
  check 58
pages
  filling percentage 187
  free 187
parallel check runs 61
password for UDS/SQL files 303
pattern 304
pattern string 304
PETA 304
physical consistency
  checking 57
physical structures
  check 57
pointer array 304
pointers
  reorganize 256, 257
PPP (probable position pointer) 305
prepared to commit (PTC) 304
primary key 304
primary key (SQL) 304
primary pages
  number 134
primary subtransaction 304
print
  act-key-0 page 218
  CALC key statistics 197
  CALC pages 225
  data pages 230
  hash area 197, 225, 228
  owner statistics 192
  realm statistics 186
  record number statistics 201
  record type-statistics 194
  Schema Information Area 119, 122
  set statistics 189
  Subschema Information Area 119, 122
print realm 214
PRINT statement
  BPRECORD 215
print table 232
PRINT-RELATIONAL-SCHEMAINFO statement
  BPSQLSIA 166
PRIVACY-AND-IQF database
  check 57
PRIVACY-AND-IQF SCHEMA 304
PRIVACY-AND-IQF SUBSCHEMA 305
probable position pointer 257
probable position pointer (PPP) 305
processing chain 305
PTC state 305
R
README file 8
READY 305
READYC 305
realm 305
  creation date 220
  number 132, 141
  number per subschema 145
  remove 33
  statistics 186
realm attach 26
realm configuration 305
realm copy 305
realm name 126, 187, 214
realm number 123, 125, 126, 134, 145, 203, 210, 219
realm reduce 273
realm reference number 305
realm size 188
modify 247
REALM statement
  BCHECK 75
  BPRECORD 214
realm statistics 178
  description 187
  print 186
realm time of last update 220
realm to be output
  specify 214
realm version
  internal 220
realm-name 17
realmref 17
realms to be checked
    specify 75
reconfiguration 306
record 151, 217, 230, 232, 306
record address 306
RECORD AREA 149, 156, 306
record displacement
    check 58
record element 306
record hierarchy 306
RECORD INFORMATION 129, 148
record number statistics 180, 201
    print 201
record population
    modify 250
record reference number 306
record SEARCH KEY table 306
record sequence number 132, 195, 212, 306
    check 58
RECORD statement
    BCHECK 76
record type 125, 307
    length 123, 129
    linear 307
    name 129, 132, 203, 228
    number 123, 128, 129, 132, 228
    number per realm 128
    schema format 150, 156, 159
    specify for checking 76
    statistics 179, 194
    subschema format 150, 156, 159
record type-statistics
    print 194
RECORD WITHIN LIST 128
record-name 17
recordref 17
records stored
    number of 203
REC-REF 306
reduce realm 273
REFERENCE NUMBERS 125, 146
referential integrity 307
relational access 161
relational schema information 161
remote application program 307
remote configuration 307
remote database 307
remote host 307
remove
    overflow pages 256, 257
    realms 33
REMOVE-REALM statement
    BMEND 33
reorganize
    CALC areas 254, 275
    pointers 256, 257
    set constructs 262
    set tables 194
    tables 191, 194, 262, 264
REORGANIZE-CALC statement
    BREORG 254
REORGANIZE-POINTERs statement
    BREORG 261
repeating group 307
request 307
RESET statement
    BMODTT 280
restart
    of a session 307
    of BMEND 307
restructuring 308
return code 308
REUSE statement
    BMODTT 279
RLOG file 308
rollback 308
RSQ 308
RUNUNIT-ID 308
S
saving 243
schema 308
    designate 213
    identify 74
    name 121, 122, 213
schema (cont.)
  specify 268
schema DDL 308
schema format 150, 151
  record type 150, 156, 159
schema information
  relational 161
Schema Information Area 117, 122
  print 119, 122
Schema Information Area (SIA) 308
SCHEMA statement
  BCHECK 74
  BPRECORD 213
schema-name 17
SCRTCH1 file 182, 239
  BCHECK 66
SDF 162
SDF statements notational conventions 14
SEARCH KEY 308
SEARCH KEY table 308
SEARCH keys to be checked
  specify 80
secondary key 309
secondary subtransaction 309
select
  checking mode 71
  consistency criterion 72
  subschemas 166
sequence number 309
sequential access 309
server task 309
session 309
  abort 309
  end 309
  interrupt 309
  start 310
Session Log File (SLF) 310
session restart 310
session segment 310
set 310
  checking mode 58
  dynamic 136, 137, 310
  implicit 129, 136, 310
  membership 136
  name 136, 228
  number 123, 136, 139, 142, 158
  singular 136, 310
  sort sequence 136
  specify for checking 78
  standard 310
  statistics 179, 189
  type 136
set connection data 129, 138, 141, 159, 215, 216, 217
  length 138
Set Connection Data (SCD) 308
set constructs
  reorganize 262
SET INFORMATION 135, 155
set mode 136
set occurrence 310
SET OCCURRENCE SELECTION 156
set occurrence selection 136
set reference number 310
set SEARCH KEY table 311
SET statement
  BCHECK 78
  BMODTT 280
set statistics
  print 189
set tables
  filling ratio 191, 193
  number of index levels 191
  reorganize 194
set-name 17
shadow database 311
  check 60
  overall check 63
Shared User buffer pool 311
SHARED-RETRIEVAL
  database operation 61
SHOW-LOG-INFORMATION statement
  BMEND 34
SIA 311
SIA PRINT REPORT 122
SIA report 122, 125
SIA VALIDATION DATE 144
SIB 311
singular set 136
size
  DBTT 195
  realm 188
size calculation
  SORTWK file 183
SLF 311
sort buffer
  define size 70
sort check 58, 59
sort key table 311
sort sequence
  set 136
sort sequence of keys
  check 58
SORTCORE statement
  BCHECK 70
sorting procedure 59
SORTWK file 183, 240
  BCHECK 66
    calculate file 183
    calculating size 183
source program 311
spanned record 74, 311
specify
  realm to be output 214
  realms to be checked 75
  record types to be checked 76
  schema 268
  SEARCH keys to be checked 80
  sets to be checked 78
  subschema 269
SPECIFY-SCHEMA statement
  BREORG 268
SPECIFY-SUBSCHEMA statement
  BREORG 269
SQL 311
SQL accesses
  base tables 171
SQL conversation 312
SQL DML 312
SQL interface 161
SQL Interface Block (SIB) 312
SQL transaction 312
SSIA 312
SSIA (Subschema Information Area)
  length 144
SSIA PRINT REPORT 144
SSIA report 144
SSIA-RECORD 312
SSITAB module 312
SSL 312
standard set 312
start logging 40
START-LOG statement
  BMEND 40
statement
  undo 46, 270
statement code 312
statement sequence
  BPRECORD 211
statements
  BMODTT 279
  BPRECORD 211
  BPSIA 119
statistics
  CALC key 180, 197
  owner 179, 193
  realm 178, 186
  record number 180, 201
  record type 179, 194
  set 179, 189
status code 312
stop logging 31
STOP-LOG statement
  BMEND 45
Storage Structure Language (SSL) 312
stored records
  number 191, 195
string 152, 313
structure
  job variable 52
subcontrol system 313
subschema 313
  designate 185
  format 150, 151
  keys 158
  name 121, 144
Index

subschema (cont.)
  specify 269
Subschema DDL 313
subschema format
  record type 150, 156, 159
Subschema Information Area 117, 144
  print 119, 122
Subschema Information Area (SSIA) 313
subschema module 313
subschema record 313
SUB-SCHEMA SECTION 313
SUBSCHEMA statement 185
  BSTATUS 185
subschema-name 18
subschemas
  select 166
subtransaction 313
summing checks 58, 59
  using internal results 68
syntax description 14
system area 313
system break information 220, 314
system buffer pools 314
system environment
  BCHECK 61, 62
  BMODTT 278
  BPRECORD 208
  BPSIA 118
  BPSQLSIA 162
  BREORG 237
  BSTATUS 181
SYSTEM record 314
SYSTEM set 136, 314

T
  table
    address 141
    CALC key 142, 228
    description 233
    entries 232
    filling ratio 191, 193
    header 232
    multi-level 314
    number of index levels 191, 194
    print 232
    reorganize 191, 194, 262, 264
table (SQL) 314
table header 314
table pages 314
  check 58
  TANGRAM 314
task attribute TP 315
task communication 315
Task Currency Area 144, 147, 148, 156
task deadlock 315
task priority 315
task switch 29
  BREORG 243
TCUA 315
terminate input 30, 164, 246
time 18
time acknowledgment 315
time of last update
  realm 220
transaction 315
  committing a 315
  roll back 315
Transaction Currency Area 123, 316
transaction identification (TA-ID) 316
transfer pool 316
two-phase commit protocol 316
type
  key item 159
  set 136
TYPE statement
  BCHECK 72

U
  UDS 316
  UDS/UTM-D consistency 317
  UDSADM 316
  UDS-D task UDSCT 316
  UDSHASH 316
  UDSNET 316
  UDSSUB 316
  UNDO statement
    BMEND 46
    BREORG 270
undo statement 46, 270
unique throughout the network 317
update job variable 55
UPDATE-DATABASE statement
   BMEND 47
used DBTT entries 195
user database 317
user realm 317
   check 57
user task 317
userid 18
USER-WORK-AREA (UWA) 148, 317
using internal results
   summing checks 68
utility routine
   BPSQLSIA 161
UTM 317
UWA 317

V
variable 12
   item 152
vector 317
version number
   internal 317
volume 18

W
warm start 318
warnings
   BCHECK 89
work files
   BCHECK 66
   BREORG 239
   BSTATUS 182

X
x-string 18
## Contents

1. Preface .................................................................................................................. 1
   1.1 Brief product description ................................................................. 1
   1.2 Target group ......................................................................................... 2
   1.3 Summary of contents ............................................................................. 3
   1.4 README file ......................................................................................... 8
   1.5 Changes since the last version ............................................................. 9
   1.6 Notational conventions ....................................................................... 12
      1.6.1 Warnings and notes .................................................................... 12
      1.6.2 NON-SDF notational conventions ............................................. 12
      1.6.3 SDF syntax representation ......................................................... 14
   1.7 Sample databases ................................................................................. 19

2. Updating and reconstructing a database with BMEND ....................... 21
   2.1 Functions ............................................................................................. 22
   2.2 Statements ........................................................................................... 23
      2.2.1 Rules ............................................................................................. 23
      2.2.2 Permitted functions ..................................................................... 24
      2.2.3 BMEND statements ..................................................................... 25
         Attach realms to a database (ADD-REALM) ........................................ 26
         Define buffer size (ALLOCATE-BUFFER-POOL) ............................... 27
         Disable online-copy facility for the database (DISABLE-ONLINE-COPY) 28
         Enable online-copy facility for the database (ENABLE-ONLINE-COPY) 29
         Terminate input of statements (END) ............................................ 30
         Stop logging for an inconsistent database (KILL-LOG) ................. 31
         Open database (OPEN-DATABASE) ............................................... 32
         Detach realms (REMOVE-REALM) .................................................. 33
         Show log information (SHOW-LOG-INFORMATION) ..................... 34
         Start logging for database original (START-LOG) ....................... 40
         Stop logging for database operations (STOP-LOG) ..................... 45
         Undo a statement (UNDO) ............................................................. 46
         Apply AFIMs to a database (UPDATE-DATABASE) ...................... 47
   2.2.4 Command sequence to start BMEND .............................................. 50
   2.3 Supplying job variables ..................................................................... 51
<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Checking the consistency of a database with BCHECK</td>
</tr>
<tr>
<td>3.1</td>
<td>Description of the checking procedure</td>
</tr>
<tr>
<td>3.1.1</td>
<td>Setting the checking mode</td>
</tr>
<tr>
<td>3.1.2</td>
<td>Defining the scope of checking</td>
</tr>
<tr>
<td>3.1.3</td>
<td>Checking for coherence</td>
</tr>
<tr>
<td>3.2</td>
<td>System environment</td>
</tr>
<tr>
<td></td>
<td>Required work files</td>
</tr>
<tr>
<td>3.3</td>
<td>Using the results of the summing run in a sort run</td>
</tr>
<tr>
<td>3.4</td>
<td>Statements for BCHECK</td>
</tr>
<tr>
<td></td>
<td>Define size of sort buffer (SORTCORE)</td>
</tr>
<tr>
<td></td>
<td>Select checking mode and define extent of checking (CHECK)</td>
</tr>
<tr>
<td></td>
<td>Select consistency criteria (TYPE)</td>
</tr>
<tr>
<td></td>
<td>Identify schema (SCHEMA NAME)</td>
</tr>
<tr>
<td></td>
<td>Specify realms to be checked (REALM NAME)</td>
</tr>
<tr>
<td></td>
<td>Specify record types to be checked (RECORD NAME)</td>
</tr>
<tr>
<td></td>
<td>Specify sets to be checked (SET NAME)</td>
</tr>
<tr>
<td></td>
<td>Specify SEARCH keys to be checked (KEY REF)</td>
</tr>
<tr>
<td>3.5</td>
<td>Command sequence to start BCHECK</td>
</tr>
<tr>
<td>3.6</td>
<td>BCHECK examples</td>
</tr>
<tr>
<td>3.7</td>
<td>Messages</td>
</tr>
<tr>
<td>3.7.1</td>
<td>Warnings</td>
</tr>
<tr>
<td>3.7.2</td>
<td>Error messages</td>
</tr>
<tr>
<td>3.7.3</td>
<td>Execution messages</td>
</tr>
<tr>
<td>3.7.4</td>
<td>Inconsistency messages</td>
</tr>
<tr>
<td>3.7.4.1</td>
<td>Global inconsistencies (without index check)</td>
</tr>
<tr>
<td>3.7.4.2</td>
<td>Global inconsistencies (index check)</td>
</tr>
<tr>
<td>3.7.4.3</td>
<td>Local inconsistencies</td>
</tr>
<tr>
<td>3.7.5</td>
<td>Usage of job switches</td>
</tr>
<tr>
<td>4</td>
<td>Printing the schema/subschema information area with BPSIA</td>
</tr>
<tr>
<td>4.1</td>
<td>System environment</td>
</tr>
<tr>
<td>4.2</td>
<td>BPSIA statements</td>
</tr>
<tr>
<td></td>
<td>Print a schema (DISPLAY SCHEMA)</td>
</tr>
<tr>
<td></td>
<td>Print a subschema (DISPLAY SUBSCHEMA)</td>
</tr>
<tr>
<td></td>
<td>Terminate statement input (END)</td>
</tr>
<tr>
<td>4.3</td>
<td>Command sequence for starting BPSIA</td>
</tr>
<tr>
<td>4.4</td>
<td>Description of the SIA report</td>
</tr>
<tr>
<td></td>
<td>SIA PRINT REPORT (general information)</td>
</tr>
<tr>
<td></td>
<td>REFERENCE NUMBERS</td>
</tr>
<tr>
<td></td>
<td>AREA INFORMATION</td>
</tr>
<tr>
<td></td>
<td>RECORD WITHIN LIST</td>
</tr>
<tr>
<td></td>
<td>RECORD INFORMATION</td>
</tr>
<tr>
<td></td>
<td>DBTT INFORMATION</td>
</tr>
<tr>
<td></td>
<td>CALC INFORMATION</td>
</tr>
</tbody>
</table>
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>SET INFORMATION</td>
<td>135</td>
</tr>
<tr>
<td>KEY INFORMATION (NO CALC SEARCH KEYS)</td>
<td>139</td>
</tr>
<tr>
<td>CALC-SEARCH-KEY INFORMATION</td>
<td>142</td>
</tr>
<tr>
<td>4.5 Description of the SSIA report</td>
<td>144</td>
</tr>
<tr>
<td>SSIA PRINT REPORT (general information)</td>
<td>144</td>
</tr>
<tr>
<td>REFERENCE NUMBERS</td>
<td>146</td>
</tr>
<tr>
<td>AREA INFORMATION</td>
<td>147</td>
</tr>
<tr>
<td>RECORD INFORMATION</td>
<td>148</td>
</tr>
<tr>
<td>CALC KEY INFORMATION</td>
<td>150</td>
</tr>
<tr>
<td>ITEM STRING LIST</td>
<td>151</td>
</tr>
<tr>
<td>KEY ITEM LIST</td>
<td>153</td>
</tr>
<tr>
<td>SET INFORMATION</td>
<td>155</td>
</tr>
<tr>
<td>KEY INFORMATION</td>
<td>158</td>
</tr>
<tr>
<td>5 Output relational schema information with BPSQLSIA</td>
<td>161</td>
</tr>
<tr>
<td>5.1 Overview</td>
<td>161</td>
</tr>
<tr>
<td>5.2 System environment</td>
<td>162</td>
</tr>
<tr>
<td>5.3 Prerequisites for SQL access to CODASYL definitions</td>
<td>163</td>
</tr>
<tr>
<td>5.4 BPSQLSIA statements</td>
<td>164</td>
</tr>
<tr>
<td>Terminate input (END)</td>
<td>164</td>
</tr>
<tr>
<td>Open database (OPEN-DATABASE)</td>
<td>165</td>
</tr>
<tr>
<td>Select subschemas (PRINT-RELATIONAL-SCHEMINFO)</td>
<td>166</td>
</tr>
<tr>
<td>5.5 Command sequence to start BPSQLSIA</td>
<td>167</td>
</tr>
<tr>
<td>5.6 Description of the output of BPSQLSIA</td>
<td>168</td>
</tr>
<tr>
<td>5.7 Conversion rules</td>
<td>169</td>
</tr>
<tr>
<td>5.8 Summary of the SQL access permitted for each base table</td>
<td>171</td>
</tr>
<tr>
<td>5.9 Example</td>
<td>172</td>
</tr>
<tr>
<td>6 Printing statistics on the occupied storage space with BSTATUS</td>
<td>177</td>
</tr>
<tr>
<td>6.1 Functions</td>
<td>178</td>
</tr>
<tr>
<td>6.2 System environment</td>
<td>181</td>
</tr>
<tr>
<td>Work files</td>
<td>182</td>
</tr>
<tr>
<td>6.3 BSTATUS statements</td>
<td>184</td>
</tr>
<tr>
<td>Designate the subschema (SUBSCHEMA)</td>
<td>185</td>
</tr>
<tr>
<td>Print realm statistics (DISPLAY REALM)</td>
<td>186</td>
</tr>
<tr>
<td>Print set statistics (DISPLAY TABLE FOR SET)</td>
<td>189</td>
</tr>
<tr>
<td>Print owner statistics (DISPLAY TABLE FOR OWNER)</td>
<td>192</td>
</tr>
<tr>
<td>Print record type statistics (DISPLAY RECORD)</td>
<td>194</td>
</tr>
<tr>
<td>Print CALC key statistics (DISPLAY CALC)</td>
<td>197</td>
</tr>
<tr>
<td>Print record number statistics (DISPLAY RECORDNUMBER)</td>
<td>201</td>
</tr>
<tr>
<td>Terminate input (END)</td>
<td>203</td>
</tr>
<tr>
<td>6.4 Command sequence to start BSTATUS</td>
<td>204</td>
</tr>
</tbody>
</table>
## Contents

### 7 Printing out the contents of realms with BPRECORD

7.1 System environment .......................... 207
7.2 General description of the output of BPRECORD ........................................... 208
7.3 BPRECORD statements .......................... 211
   Designate the schema (SCHEMA NAME) ........................................ 213
   Specify the realm to be printed (REALM NAME) .................................. 214
   Determine scope of output (PRINT) .............................................. 215
   Print act-key-0 page (DISPLAY PAGE) ........................................ 218
   List FPA entries (DISPLAY FPA) .................................................. 222
   List DBTT entries (DISPLAY DBTT) ................................................ 223
   Print CALC pages (DISPLAY CALC) ................................................ 225
   Print data pages (DISPLAY DATA) .................................................. 230
   Terminate BPRECORD (END) ....................................................... 233
7.4 Command sequence to start BPRECORD ................................................. 234

### 8 Database reorganization with BREORG

8.1 Functions .......................................... 236
8.2 System environment .................................. 237
   Work files ................................................................. 239
8.3 Database saving ...................................... 243
8.4 BREORG statements ................................... 244
   Define buffer size (ALLOCATE-BUFFERPOOL) .................................. 245
   Terminate input of statements (END) ......................................... 246
   Modify realm size (MODIFY-REALM-SIZE) .................................... 247
   Modify record population (MODIFY-RECORD-POPULATION) .................. 250
   Open database (OPEN-DATABASE) ............................................. 253
   Reorganize CALC areas (REORGANIZE-CALC) ................................ 254
   Reorganize all PPPs in a realm (REORGANIZE-POINTERS) .................... 261
   Reorganize tables and set constructs (REORGANIZE-SET) ................. 262
   Specify schema (SPECIFY-SCHEMA) ......................................... 268
   Specify subschema (SPECIFY-SUBSCHEMA) .................................... 269
   Undo statement (UNDO) ......................................................... 270
8.5 Command sequence to start BREORG .................................................. 271
8.6 Examples .............................................. 273

### 9 Controlling the reuse of database keys and the free place search with BMODTT

9.1 System environment .................................. 278
9.2 BMODTT statements ................................... 279
9.3 Command sequence to start BMODTT ............................................. 281

Glossary .................................................... 283

Abbreviations ............................................. 319
Contents

Related publications ......................................................... 323
Index ............................................................................. 339
UDS/SQL (BS2000/OSD)

Recovery, Information and Reorganization

Target group
This manual is intended for the database administrator

Contents
This manual describes activities required for the administration and operation of a database. These include updating, reconstruction, consistency checking, outputting information, and reorganization.

Edition: November 2004

File: uds_sir.pdf


All rights reserved.
Delivery subject to availability; right of technical modifications reserved.

All hardware and software names used are trademarks of their respective manufacturers.

This manual was produced by cognitas. Gesellschaft für Technik-Dokumentation mbH
www.cognitas.de
Information on this document

On April 1, 2009, Fujitsu became the sole owner of Fujitsu Siemens Computers. This new subsidiary of Fujitsu has been renamed Fujitsu Technology Solutions.

This document from the document archive refers to a product version which was released a considerable time ago or which is no longer marketed.

Please note that all company references and copyrights in this document have been legally transferred to Fujitsu Technology Solutions.

Contact and support addresses will now be offered by Fujitsu Technology Solutions and have the format …@ts.fujitsu.com.


Copyright Fujitsu Technology Solutions, 2009

Hinweise zum vorliegenden Dokument


Das vorliegende Dokument aus dem Dokumentenarchiv bezieht sich auf eine bereits vor längerer Zeit freigegebene oder nicht mehr im Vertrieb befindliche Produktversion.

Bitte beachten Sie, dass alle Firmenbezüge und Copyrights im vorliegenden Dokument rechtlich auf Fujitsu Technology Solutions übergegangen sind.


Copyright Fujitsu Technology Solutions, 2009